Visual Analysis of Multimodal Sensor Data

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Computer Science)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Joao Rulff

August 2025

Visual Analysis of Multimodal Sensor Data

THESIS

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Computer Science)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Joao Rulff

August 2025

Approved:

Department Chair Signature

Date

Approved by the Guidance Committee:

Major: PhD in Computer Science

Claudio T. Silva
Institute Professor
NYU Tandon School of Engineering

Date

Juliana Freire
Institute Professor
NYU Tandon School of Engineering

Date

Maryam Hosseini
Assistant Professor
University of California Berkeley

Date

Robert Krueger
Assistant Professor
NYU Tandon School of Engineering

Date

Vita

Joao Rulff was born in the city of Rio de Janeiro, Brazil. He has a B.Sc. in Computer Science from Universidade Federal Fluminense. While completing his undergraduate studies, he was a visiting student at Monmouth University and completed internships at SLAC-Stanford, IBM, and STI-UFF. He started his Ph.D. in the fall of 2019, working in data visualization and human-computer interaction. Throughout his PhD, he has received an urban doctoral fellowship to continue his work on urban data analytics. He has published papers on top-tier venues, such as IEEE VIS, ACM CHI, and Eurovis. He was also awarded two Best Paper Honorable Mention awards at IEEE VIS 2023 and SIBGRAPI 2023. During his PhD he completed internships at Capital One and NEC Labs

Acknowledgements

This dissertation would not be possible without the unconditional support from my family. My mother and Father, Bernadete and Joao, my sister, Clara, and my partner, Erika, were fundamental during times of uncertainty throughout my Ph.D. Especially during the global COVID-19 pandemic, when we could not see each other in person for almost two years.

I would like to especially thank my advisor, Claudio Silva, for the opportunity to join his group at New York University, the continuous support and guidance throughout my Ph.D. studies, and the chance to work on incredible projects. I would also like to thank the members of my committee, Juliana Freire, Maryam Hosseini, and Robert Krueger, for their valuable feedback.

I could not forget to express my gratitude to Marcos Lage, my undergraduate advisor, who introduced me to the academic world and supported and guided me in every decision I had to make throughout the past 10 years. Marcos and Lhaylla Crissaff were fundamental pieces in my journey, transcending the professional relationship and becoming close friends. I would also like to acknowledge my undergraduate thesis advisor, Vanessa Braganholo.

The work I was able to do during my Ph.D. was only possible due to the amazing researchers I had the chance to learn from. I would like to thank Fabio Miranda, Harish Doraiswamy, Luis Gustavo Nonato, Brian Barr, Qi Sun, Michael Krone, Huy T. Vo, Mark Cartwright, Graham Dove, Charlie Mydlarz, Magdalena Fuentes, and Juan Bello.

All the friends at VIDA, some of them collaborators, also played a pivotal role during these years. They made the process easier and way more fun. To Aecio Santos, Juliana Barbosa, Fernando Chirigati, Raoni Lourenco, Jonathas Costa, Peter Xenopoulos, Guande Wu, Shaoyu Chen, Sonia Castelo, Roque Lopez, Giancarlo Pereira, Parikshit Solunke, Erin McGowan, Iran Roman, Bea Steers, Jorge Ono, Yurii Piadyk, Remi Rampin, Vicky Rampin, Grace Fan, Christos Koutras, Eduardo Pena, Joao Fonseca, Andrew Bell, Fabio Felix, and Caterina Fuligni, my deepest thank you.

Last but not least, I would like to thank all the exceptional professors I had the chance to take courses from at Tandon School of Engineering and the Center for

Data Science. The same recognition goes to the NYU admin team: Ann Borray, Kari Schwartz, Eve Henderson, and Susana Garcia.

For my family, with all my love and gratitude.

ABSTRACT

Visual Analysis of Multimodal Sensor Data

by

Joao Rulff

Advisor: Prof. Claudio Silva, Ph.D.

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy (Computer Science)

August 2025

The proliferation of low-cost, high-fidelity sensors deployed in multiple contexts, such as in wearable technology and urban infrastructure, has led to an unprecedented deluge of multimodal data, capturing complex real-world phenomena from different perspectives. While this data holds the potential for profound insights into human behavior and urban dynamics, its sheer volume, variety, and complexity present significant challenges for analysis. Raw sensor readings from multimodal data, such as video, audio, and time-series streams, are often opaque and require specialized tools to transform them into actionable knowledge for domain experts. This thesis addresses the need for new analytical systems by presenting a series of novel interactive frameworks designed to support the exploration and understanding of multimodal, sensor-acquired data. These systems bridge the gap between large-scale, heterogeneous datasets and human cognition through a synergistic combination of intelligent data management, machine learning, and interactive visualization. This thesis starts by presenting the role of data captured by sensors mounted on wearable headsets in supporting the debugging of immersive intelligent assistants in ARGUS. The focus then shifts to urban analytics, where the thesis introduces a two-part contribution to better understand pedestrian activity at street intersections. First, it presents the StreetAware Dataset, a novel, high-resolution, synchronized multimodal dataset featuring multi-perspective video, audio, and LiDAR scans. Leveraging this unique resource, the thesis then introduces Crossroads, a pedestrian-centric visual analytics system that analyzes safety patterns using an automatic data enrichment pipeline and an interactive human-in-the-loop workflow. Finally, this work presents Urban Rhapsody, a framework for the large-scale exploration of urban soundscapes that integrates machine listening with an interactive interface for querying and building custom classification models. Together, these frameworks demonstrate the power of visual analytics to make complex sensor data accessible, interpretable, and actionable, providing critical tools for researchers and practitioners in domains ranging from immersive analytics to urban planning and transportation safety.

Table of Contents

List of Figures

Chapter 1

Introduction

We live in an era of unprecedented data proliferation, offering a granular, digital lens into previously unexplored phenomena. From the macro-scale of city-wide activity driven by citizens to the micro-scale of humans performing everyday manual tasks, sensors are capturing a deluge of information that holds the potential to unlock profound insights into complex real-world dynamics. This trend is largely driven by the democratization of sensing technology. Significant advancements in hardware downsizing and a sharp decline in manufacturing costs are enabling broader access to these devices. The deployment of low-cost, high-fidelity sensors is no longer confined to specialized instrumented environments; instead, they are integrated into our cities, devices, and even wearable technologies, creating a digital network that constantly monitors and records the world around us.

The data produced by these sensing initiatives is inherently multimodal, capturing phenomena from multiple perspectives and through various signal types. Sensors integrated into wearable devices, such as augmented reality headsets, provide an egocentric or first-person view, capturing data streams including egocentric video, gaze tracking, and inertial measurement unit (IMU) data, which offer intimate insights into human actions and attention. Conversely, sensors deployed on city infrastructure provide an exocentric perspective, yielding vast datasets such as multi-view synchronized videos of intersections and longitudinal audio recordings of the urban soundscape. This heterogeneity in data sources provides a rich, multi-faceted view of real-world activities but also introduces significant challenges for data management and analysis.

Although valuable, the sheer amount of data collected by multimodal sensors can only be truly useful when domain experts—be they specialists in specific domains, such as urban planners, transportation engineers, or immersive technology developers—are equipped with the proper tools to navigate and explore it. Raw sensor readings, in their unprocessed state, are often opaque and hard to interpret. To transform this data into meaningful insights and actionable knowledge, there is a critical need for analytical systems that can bridge the gap between complex, large-scale datasets and human cognition. Such tools must enable experts to explore, query, summarize, and visualize multimodal information, allowing them to uncover patterns, validate hypotheses, and debug system behaviors that would otherwise remain buried in the data. Without these instruments for analysis, the potential held within our sensor-rich world remains largely untapped.

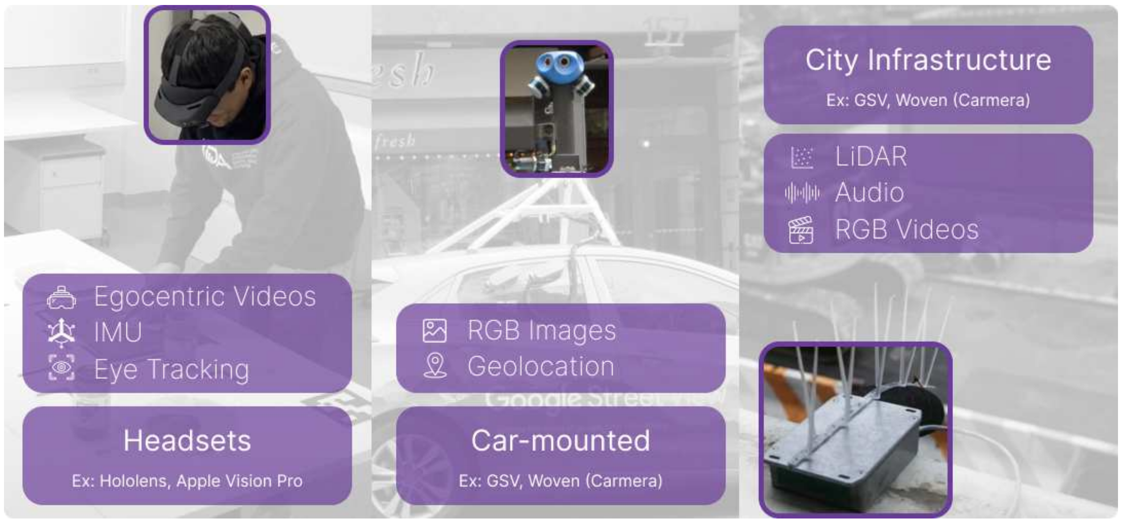


Figure 1.1: Examples of data collected by sensors mounted on different environments. Sensors mounted on immersive headsets can capture data representing human actions, such as egocentric videos and eye tracking. Sensors mounted on cars, can capture geolocated images representing different city neighborhoods. Lastly, static sensors deployed on the city infrastructure can capture temporally-dense videos and audio recordings of pedestrian activity.

Building such visual analysis systems, however, presents its own set of challenges. It requires a synergistic combination of smart data management techniques and visualization approaches. Handling the sheer volume, velocity, and variety of the data demands architectures to ingest, store, and index terabytes of heterogeneous data—from high-resolution video streams to high-frequency audio recordings—in a way that supports interactive query times, a prerequisite for fluid, exploratory analysis. Also, regarding interactivity and visualization, there is a need to develop novel techniques that move beyond conventional charts and graphs. These methods must be capable of summarizing complex spatiotemporal patterns and providing intuitive interfaces for experts to compose expressive queries across different modalities. The ultimate goal is to create integrated environments that fuse scalable data processing with powerful visual metaphors, enabling the interplay between computational analysis and human expertise.

The research described in this thesis proposal presents new interactive frameworks to support domain specialists in better understanding and exploring multimodal, sensor-acquired data. These frameworks range from supporting the debugging of complex human-AI interactions to the analysis of urban activities, such as pedestrian behavior and noise pollution. These systems are composed of various modules, such as processing pipelines to extract expressive information from collections of audio and video, and information-rich interfaces to support meaningful exploration of large datasets through sophisticated query capabilities and data summarization. The main contributions are described below:

1. **Interactive debugging of intelligent assistants.** Debugging immersive intelligent assistant systems is a complex task, as it requires developers to analyze multiple data streams from sensors and machine learning (ML) models that must work together in real-time. To address this challenge, ARGUS provides a visual analytics system designed to help developers troubleshoot, improve, and fine-tune the components of an AR assistant. The system's primary contribution is its dual-mode functionality, which supports both live, online monitoring during task execution and offline, retrospective analysis of historical data. ARGUS introduces novel visual representations to analyze these complex spatiotemporal and multimodal data streams, allowing developers to uncover interaction patterns between a performer's actions and the ML model outputs. The paper demonstrates the system's usefulness through case studies where developers leverage the tool to improve their systems.

2. **Novel dataset to support multimodal analysis of intersection dy-**

namics. To better understand pedestrian dynamics at intersections, we introduced StreetAware. StreetAware is a novel, high-resolution, synchronized, multimodal dataset designed for analyzing urban intersection dynamics. Captured using custom REIP sensors at three intersections in Brooklyn, New York, the dataset provides nearly 8 hours of accurately synchronized, multi-perspective data, including high-resolution video, multiple audio channels, and LiDAR scans, all fully anonymized. StreetAware aims to address the lack of comprehensive data for studying complex interactions between vehicles, pedestrians, and the environment, facilitating research in computer vision, urban sensing, and smart city applications like accessibility-aware design and Vision Zero initiatives.

3. **Interactive analysis of pedestrian mobility.** CrossRoads is a pedestrian-centric visual analytics system designed to analyze the complex dynamics and safety patterns at urban intersections using multimodal data, integrating video, audio, and traffic signal information. Addressing the limitations of traditional vehicle-focused analyses and the challenges of processing noisy, real-world footage, CrossRoads employs an automatic data enrichment pipeline to extract semantic information like agent trajectories (pedestrians, cyclists, vehicles) and audio events (honks, sirens). It uniquely features a human-in-the-loop approach for users to refine and validate 3D trajectory reconstructions, enhancing accuracy. Through a set of interactive, linked visualizations, the tool supports multiscale exploration—from global intersection statistics and movement patterns to detailed inspection of specific incidents like near-misses—enabling users to perform expressive queries across modalities, identify high-risk behaviors, and ultimately inform evidence-based safety interventions and infrastructure design.

4. **Longitudinal exploration of city soundscapes.** First, we introduce Urban Rhapsody, a visual analytics framework designed for exploring large-scale urban audio datasets. It integrates machine listening algorithms for audio representation with an interactive visual interface, allowing users to query by audio sample, interactively label sounds based on their perception, and iteratively build custom classification models (called ”prototypes”) for

complex sound concepts. Through linked visualizations including maps, temporal views, multidimensional projections, and spectrograms, the system enables the analysis of sound patterns, mixtures, and anomalies across space and time, facilitating a deeper understanding of the urban soundscape beyond simple noise level measurements.

1.0.1 Organization

The remainder of this thesis is structured as follows. First, chapter 2 describes ARGUS, a system that leverages sensor-acquired data to allow for the debugging of human-ai interactions in a task-guidance setting. Next, we proceed to describe the efforts in creating a framework to facilitate the analysis of pedestrians interacting with urban environments. Chapter 3 introduces StreetAware, a multimodal and multiview dataset comprising synchronized audio and video streams that represent intersection activity in New York City. Then, Chapter 4 presents Crossroads, a visual analytics system designed to support the exploration of street-level multimodal data, helping domain experts categorize intersection activity. Lastly, Chapter 5 shows Urban Rhapsody, a framework to support the large-scale exploration of urban soundscapes. This thesis concludes, in Chapter 6, with a discussion on the need for more reproducible practices in developing visual analytics systems and presents an initial prototype of a library to facilitate the development of interactive and spatiotemporal visualizations.

Chapter 2

ARGUS: Visualization of AI-Assisted Task Guidance in AR

2.1 Introduction

The concept of an augmented reality (AR) assistant has captured the human imagination for years, becoming a staple of modern science fiction through popular franchises such as the *Marvel Cinematic Universe*, *Star Trek*, and *Terminator*. The applications of such a system are seemingly endless. Humans, even those with domain expertise, are fallible creatures with imperfect memories whose skills deteriorate over time, especially during repetitive tasks or under stress. An AR assistant could help experts and novices alike in performing both familiar and new tasks. For instance, an AR assistant could aid a surgeon performing a familiar yet complex procedure, who could benefit from a second set of "eyes" due to the high-stakes nature of their task. Equally, it could walk an amateur chef through the steps of an unfamiliar recipe. In an ideal scenario, the AR assistant would become "invisible" in the sense that it is seamlessly integrated into the task procedure, providing well-timed audio and visual feedback to guide uncertain performers and correct human errors while otherwise fading into the background. Overall, the AR assistant would be able to reduce human error via correction, improve performance by reducing cognitive load, and introduce new tasks across a wide variety of applications.

While aspects of this vision are currently still aspirational, we are finally beginning to develop the technology that allows concepts once relegated to the world of science fiction to become reality. With respect to machine perception, the recent explosion of research on machine learning (ML), especially deep neural networks, has given way to powerful models able to detect objects, actions, and speech in real time with high accuracy. Ever-evolving implementations of Bayesian neural networks, reinforcement learning, and dialog systems (e.g., conversational agents) allow for task modeling and transactional question answering. A rise in AR technology, especially the commercial availability of headsets such as Microsoft HoloLens 2, Magic Leap, Google Glass, or Meta Quest Pro (and soon, Apple Vision Pro) has provided the hardware necessary for task guidance. The time is ripe for the development of assistive AR systems.

Challenges in perceptually-enabled task guidance Developing an AR assistant, however, comes with a host of challenges. Such a system requires several moving parts to work in tandem to perceive the performer's environment and actions, reason through the consequences of a given action, and interact with both the performer and the user (for the sake of clarity, we will refer to subjects using the AR system to perform tasks during a session as "performers" and subjects using ARGUSto collect and analyze data as "users"). Creating these parts is a complex and resource-intensive process. The challenges include collecting, storing, and accessing a large volume of annotated data for model training, real-time sensor data processing for action and object recognition (or reasoning), and performer behavior modeling based on first-person perspective data collected by the AR headset (see Sec. 2.4 for a more detailed discussion of tasks and requirements).

Our Approach We propose **ARGUS: Augmented Reality Guidance and User-modeling System**, a visual analytics tool that facilitates multimodal data collection, enables modeling of the physical state of the environment and performer behavior, and allows for retrospective analysis and debugging of historical data generated by the AR sensors and ML models that support task guidance. Our tool operates in two main modes. The online mode (see Sec. 2.5.1) supports real-time monitoring of model behavior and data acquisition during task execution time. This mode displays tailored visuals of real-time model outputs, which allows users of ARGUSto monitor the system during live sessions and facilitates online debugging. Data is saved incrementally. Once finalized, all data and associated metadata collected during the task is seamlessly stored to permanent data store with analytical capabilities able to handle both structured data generated by ML models and multimedia data (e.g. video, depth, and audio streams) collected by the headset.

Our system can be used to explore and analyze historical session data by interacting with visualizations that summarize spatiotemporal information as well as highlight detailed information regarding model performance, performer behavior, and the physical environment (see Sec. 2.5.2).

Our design was inspired by requirements from developers of AR systems and experts that create and evaluate these systems in the context of the Defense Advanced Research Projects Agency's (DARPA) Perceptually-enabled Task Guidance (PTG) program [52]. These experts use ARGUSand have provided feedback throughout its development. In summary, **our main contributions are:**

- ARGUS, a visual analytics tool tailored to the development and debugging of intelligent assistive AR systems. It supports online monitoring during task execution as well as retrospective analysis and debugging of historical data by coupling a scalable data management framework with a novel multimodal visualization interface capable of uncovering interaction patterns between performer actions and model outputs.
- The design of novel visual representations to support complex spatiotemporal analysis of heterogeneous, multi-resolution data (i.e., data streams with different frame rates). ARGUSnot only supports the visualization of internal AR assistant ML states in the context of the actions of the performer, but also the visualization of the interactions of the performer

with the physical environment.

- We demonstrate the usefulness of ARGUSby a set of case studies that demonstrate real-world use of ARGUS, exhibiting how AR assistant developers leverage the tool to improve their systems.

This chapter is organized as follows: Sec. 2.2 reviews the relevant literature on assistive AR systems and visualization of related data. Sec. 2.3 provides background and context for ARGUS, including the AR personal assistant framework and architecture it is designed upon. Sec. 2.4 specifies the requirements we aim to achieve. Sec. 2.5 describes ARGUSin detail, including all components of its online real-time debugging mode and offline data analytics mode. Sec. 2.6 explores two case studies in which ARGUSproves useful to AR task assistant developers, ending with user feedback and limitations of our system. Finally, we offer concluding remarks and future work in Sec. 2.7.

2.2 Related Work

2.2.1 Assistive AR Systems

The idea of using AR technologies to build assistive systems that have an internal model of the real world and are able to augment what a performer sees with virtual content dates back more than three decades [39]. Yet only recent advances in AR display technologies and artificial intelligence (AI), combined with the processing power to run the necessary computations in real time, have enabled us to start building such systems. Referring to the terminology introduced by Milgram and Kishino [148] in their seminal paper on Mixed Reality, this not only requires a *class 3 display*—a head-mounted display (HMD) equipped with see-through capability that can overlay virtual content on top of the real world—but also a great *extent of world knowledge*. That is, the environment should be modeled as completely as possible so that the assistive system can react to objects and actions in the real world. Simultaneously, the *reproduction fidelity* and the *extent of presence* of an assistive system should be minimal, since the performer needs to focus on the real world, not be immersed in virtual content. In addition, the in-situ instructions help to reduce errors and facilitate procedural tasks. To date, results are mixed for task completion time using an assistive AR system versus not, with several studies finding longer times with assistive AR systems [226, 269] whereas others find the opposite [81]. Nevertheless, most studies agree that AR helps to reduce errors and overall cognitive load as it provides in-situ instruction and guidance.

AR can be enabled by a multitude of different display technologies, ranging from handheld devices like smartphones and tablets to projector-based solutions and heads-up displays found in airplanes or modern cars. We, however, focus on see-through AR HMDs for assistive AR systems, since these do not significantly encumber the performer. These headset displays do not restrict performers to a limited space and leave their hands free to execute situated tasks in the real world. Furthermore, they usually offer a wider range of built-in sensors for modeling the environment and performer such as cameras, microphones, or IMUs. See-through AR headset displays available today include Microsoft HoloLens 2 (the hardware platform used in our work) and Magic Leap 2.

As was proposed by Caudell and Mizell [39], a common use case for such systems is to support performers in repair and maintenance tasks [72, 100]. Similarly, AR assistants were proposed for manufacturing, e.g., training [128] or live monitoring of production lines [17]. Another prominent area for AR assistive systems is healthcare and medicine [16], e.g., to assist surgery [188] or other procedures [108, 222]. Furthermore, digital assistants can also make use of AR to enable a virtual embodiment of the assistant [116, 177, 204]. Most of the modern systems mentioned above integrate ML methods for specific tasks, e.g., for object or voice command recognition. However, they are mostly tailored to specific tasks and only have limited support for situated performer modeling and perceptual grounding. Integrating more complex AI methods will make the development and testing of such systems also more challenging.

To support the development of AR assistants, software toolkits have been proposed, for example, RagRug [76], which is designed for situated analysis, or Data visualizations in eXtended Reality (DXR) [210], which is specifically designed to build immersive analytics [139] applications. However, while such toolkits make it easier to develop feature-rich assistive systems that use data from the multiple sensors provided by the AR headset display and integrate AI methods, they do not offer explicit tools for external debugging of the required ML models and sensor streams. Our goal is to fill this gap with ARGUS. This requires visualizing the multiple data streams from the sensors as well as the output of the models.

2.2.2 Visualization of Multivariate Temporal Data

The visualization of multivariate temporal data is a very active field of research. A plethora of different methods and tools have been proposed which, for example, use multiple views, aggregation, and level-of-detail visualizations to represent the data efficiently. A review of these methods is beyond the scope of this paper, therefore, we refer to a number of comprehensive surveys [6, 114, 129].

There have been recent attempts to develop visualization systems to debug and understand the data acquired by multimodal, integrative-AI applications. PSI Studio, a platform to support the visualization of multimodal data streams [24] is able to provide useful visualization of sets of recorded sessions. However, it requires the user to not only compose their own visual interfaces by organizing predefined elements in a visualization canvas, but also to structure the streaming data in a predefined format, *psi-store*. Built with a similar goal, Foxglove [79] requires developers to organize their data into a Robot Operating System (ROS) environment. Moreover, these tools focus on supporting the visualization of the data streams and are not able to summarize long periods of recordings with visualizations. To the best of our knowledge, existing tools also lack the ability to debug associated ML models. Other visualization tools, such as Manifold [262], are tailored to the interpretation and debugging of general ML models. In our case, we are interested in a narrower set of ML models, those that pertain to the understanding the behavior of AI assistants, which have different requirements than other visualization systems.

2.3 Background: Building The TIM Personal Assistant

In this section we describe the context of the development of ARGUS. This includes the ecosystem of components needed to support intelligent AR assistant systems, ranging from software running on the headset device to data management modules able to ingest data in real-time.

2.3.1 Motivating Context

The development of ARGUSis driven in large part by the requirements of the DARPA PTG program [52]. PTG aims to develop AI technologies that help users perform complex physical tasks while making them both more versatile by expanding their skillset and more proficient by reducing their errors.

Specifically, the program seeks to develop methods, techniques, and technology for AI assistants that provide just-in-time visual and audio feedback to help with task execution. The goal is to utilize wearable sensors (head-mounted cameras and microphones) that allow the AR assistant to see what the performer sees and hear what they hear, so that the assistant can provide helpful feedback to the performer through speech and aligned graphics. The assistants learn about tasks by ingesting knowledge from checklists, illustrated manuals, training videos, and other sources of information (e.g., making a meal from a recipe, applying a tourniquet from directions, conducting a preflight check from a checklist). They then combine this task knowledge with a perceptual model of the environment to support mixed-initiative and task-focused performer dialogs. The dialogs may assist a performer in completing a task, identifying and correcting errors during a task, and instructing them through a new task, taking into consideration the performer's level of expertise. As part of PTG, our team has been building TIM, the Transparent, Interpretable, and Multimodal AR Personal Assistant, which is described below.

2.3.2 Overview of the TIM Personal Assistant

Our assistive AR framework (TIM) integrates perceptual grounding, attention, and user modeling during real-time AR tasks and is composed of multiple software and hardware components. TIM perceives the environment, including the state of the human performer, by using a variety of data streams (details below), which are the input to the task guidance system. TIM communicates with the performer through the HoloLens 2 headset display.

The task guidance system is primarily composed of three AI components that interpret the incoming data streams: (1) *Perceptual Grounding* utilizes information from historical instances of actions from similar tasks and makes its best prediction of what the current action and objects are. (2) *Perceptual Attention* takes the objects and transforms them into 3D coordinates and contextualizes objects over time in the 3D environment. (3) *Reasoning* then uses the objects and actions returned by perception to identify which step of the task the user is in and to understand whether or not they are performing the task correctly. Any of these data can be ingested and displayed on our platform, ARGUS.

2.3.3 System Architecture

Since the computational resources on HoloLens 2 are limited, TIM is implemented as a client-server architecture. To enable data streaming capabilities, the system utilizes server-side infrastructure that provides a centralized data communication hub and real-time ML inference to facilitate ingesting, operating over, and contextualizing the produced data streams. A system diagram can be seen in Fig. 2.1.

Data Orchestration and Storage The core of our architecture is Redis Streams, which we use as our data message queue. A REST + Websocket API provides a uniform abstraction layer for components to interact with. The HoloLens streams its sensor data to the API where it is made available to all other components in the system. The user is able to record data streaming sessions, which will listen and copy all data streams to disk. Later, users can

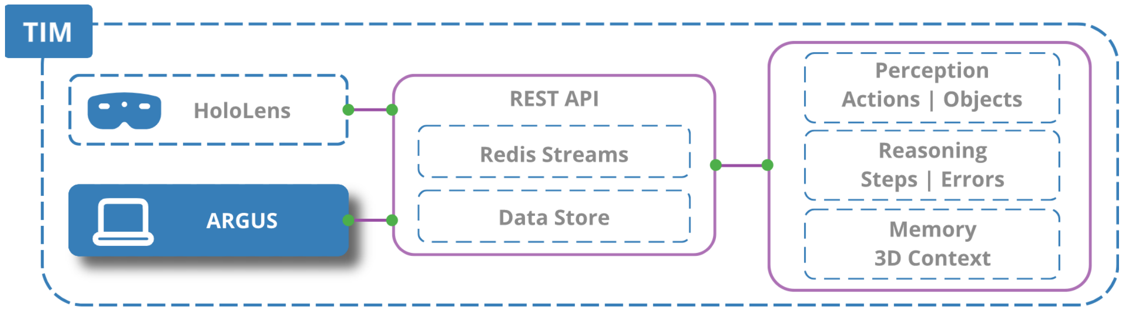


Figure 2.1: TIM's architecture proposes a data communication service between the system components: the Hololens, AI modules, and ARGUS.

selectively replay that data in the system as if the HoloLens were running, for easy offline testing.

Communication TIM uses the REST API to stream onboard sensor data (i.e., gaze, hand tracking; see details in Section 2.3.4) in real-time. This allows us to shift the computation-heavy tasks to the server while keeping the essential tasks on the HoloLens to improve responsiveness. TIM also collects the ML prediction results from the server and updates the AR interaction and interface accordingly. The AR client running on the HoloLens ingests two streams to support contextual interaction: a *perception* stream that recognizes objects in the scene and a *reasoning* stream that recognizes performer actions. On average, these streams take about 100 ms to complete one update cycle to the AR client.

2.3.4 Data

The HoloLens 2 can provide various data from multiple sensors. With *Research Mode* enabled [232], we stream data from the main RGB camera, 4 grayscale cameras, an infrared depth camera, and an IMU that contains an accelerometer, gyroscope, and magnetometer. Details of the camera data can be found in Table 2.1. Although it is theoretically possible to stream some of the data at higher resolution or frame rate, the need to run a user interface on the HoloLens creates a practical limit. Not only are the computational resources limited, but streaming extra data consumes more energy and may

Table 2.1: Description of streamed data from HoloLens 2 visual sensors.

Sensor	Resolution	Format	Framerate
RGB camera	760 × 428	RGB8	7.5 fps
Grayscale camera	640 × 480	Grayscale8	1 fps
Infrared active brightness	320 × 288	Grayscale8	5 fps
Infrared depth	320 × 288	Depth16	5 fps

result in headset overheating.

The streamed frame rate in practice may be lower due to the packet drop during streaming. Hand tracking and eye tracking data are also streamed. The eye-tracking data consists of 3D gaze origin positions and directions. The hand tracking data consists of 26 joint points for each hand. Each joint point contains a 3D position and a quaternion that indicates the orientation. In our system, the per-frame point cloud which consists of RGB and depth frames can be integrated into a holistic 3D environment. Performer sessions can vary in size. For instance, the recording of a simple recipe (preparing pinwheels [157]) usually takes ∼6 min and results in ∼600 MB data without the point cloud data, but 3 GB with the point cloud data.

Privacy and Ethical Considerations While AR provides incredible opportunities, performer privacy must be protected during data collection and utilization [183].

Our experiment protocol is approved by an Institutional Review Board (IRB). It ensures data is never directly linked to an individual identity, code numbers, rather than names or other identifying information, are used for video recordings in ARGUS. Names or any other identifiable information are not collected and do not appear in any part of the system. Despite these efforts, it is theoretically possible to re-identify performers, see, e.g., [165], where it is shown that motion data can be used for identification. Another path to re-identification is the audio produced by the voice interactions.

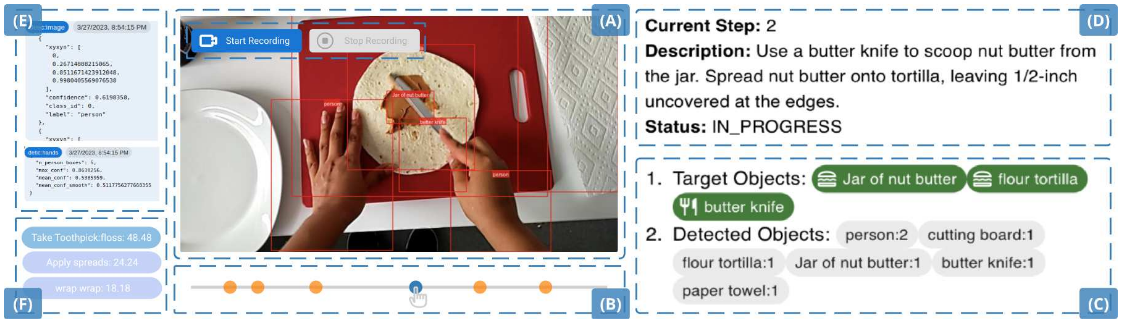
2.3.5 AI Task Guidance System

Perceptual Grounding To connect what the HoloLens sees and hears to task knowledge, the AR assistant needs to be equipped with models to recognize physical objects, actions, sounds, and contexts needed to complete a specific task. TIM uses multimodal machine-sensing models to detect human-object interactions in the environment. The output is real-time estimations with model confidence levels of three environmental elements: object categories, object localizations, and human action detections. We modulate object outputs via text instructions, allowing us to selectively detect objects and actions that are part of a particular procedure (e.g., recipe) and disregard everything else. To achieve this, our models generate and compare text and sensor representations. The models have the following main features:

Object detection and localization. We use "Detector with image classes" (Detic) [274] to generate these estimations, since it is a model that produces RGB frames and free-form text descriptions of objects of interest (e.g., "the blue cup") with bounding-box and object mask estimations for the regions in the frame where the objects are detected. Its direct comparison of RGB and text modalities is enabled via Contrastive Language-Image Pretraining (CLIP) [193].

Action recognition. TIM supports three action-recognition models: Omnivore [87], SlowFast, [69, 113, 249] and EgoVLP [189]. These models process video streams to output verb-noun tuples that describe actions. Each model has its benefits and limitations. While Omnivore is considered state-of-the-art for action recognition, it is a classification model with a fixed vocabulary. EgoVLP has a joint RGB-text representation that, similar to Detic and CLIP, allows for the detection of free-form text descriptions of actions. SlowFast integrates audio and RGB information, potentially allowing for the detection of actions outside the RGB field of view. Therefore, the optimal model to use is dependent on the deployment conditions.

Reasoning and Knowledge Transfer Reasoning and knowledge transfer first preprocess the input task description and create the corresponding objects and actions needed for each step [125, 266]. In each frame, it takes the object and action outputs from the perceptual component, along with the processed input task description, and makes two decisions. First, the reasoning module performs *error detection*, in which it attempts to determine if the performer has made an error in the current frame based on how much the objects and actions detected through perception align with the preprogrammed knowledge of the step. Second, it performs *step prediction*, in which the system predicts whether the current step is complete and should move to the next step. This decision is governed by a hidden Markov model (HMM) [15]-like approach that primarily uses the probability of each action to appear in a given step of the task. These probabilities are calculated beforehand on a training dataset.



```
{
  "xyxyn": [
    0,
    0.26714888215065,
    0.8511671423912048,
    0.9980405569076538
  ],
  "confidence": 0.6198358,
  "class_id": 0,
  "label": "person"
},
{
  "xyxyn": [
```

```
 "n_person_boxes": 5,
 "max_conf": 0.8630256,
 "mean_conf": 0.5385959,
 "mean_conf_smooth": 0.5117756277668355
}
```

Take Toothpick:floss: 48.48

Apply spreads: 24.24

wrap wrap: 18.18

Current Step: 2
Description: Use a butter knife to scoop nut butter from the jar. Spread nut butter onto tortilla, leaving 1/2-inch uncovered at the edges.
Status: IN_PROGRESS

1. Target Objects: Jar of nut butter | flour tortilla | butter knife
2. Detected Objects: person:2 | cutting board:1 | flour tortilla:1 | Jar of nut butter:1 | butter knife:1 | paper towel:1

Figure 2.2: The online component of ARGUSfor real-time debugging. (A) Streaming Video Player: Users can inspect the output of the headset's camera, overlayed with bounding boxes representing the detected objects. Users have the option to record any session. (B) Confidence Controller: a slider that allows the user to control the threshold model confidence. (C) Perception model outputs, including target and detected objects. "Target Objects" represent the objects needed in the current step (from recipe instructions) while "Detected Objects" shows all the objects identified by the perception models and their corresponding number of instances (e.g., multiple knives may be detected in a frame). (D) Reasoning model outputs, including the step and error predictions, step description, and the performer's status. (E) Raw data views show the raw data collected by the system. (F) Widgets showing the predicted actions with their probabilities. In this example, the model predicts that the current action is "Take Toothpick" with 48% likelihood, followed by "Apply Spreads" with 24% and "Wrap Wrap" with 18%.

2.4 Tasks & Requirements

ARGUSwas developed to support the development and operation of the AR personal assistant outlined in Sec. 2.3. On top of the obvious need to visualize the multitude of raw data streams, ARGUSwas designed to enable the real-time and post-hoc visualization of ML models and performer interactions in the context of the physical environment, all in a time-synchronized fashion. To summarize, such a system should have the following design requirements (R1-R5). These requirements were created by working side-by-side with the developers of the AR assistant components described in Section 2.3 (i.e., perception, reasoning), and with end users through interviews and feedback sessions during and after use of TIM and ARGUS. For context, the AR-enabled tasks that ARGUSaims to support include, but are not limited to: making a meal from a recipe, applying a tourniquet, repairing an engine, and completing an aircraft preflight check.

[**R1**] **Live monitoring**: The ability to visualize the output of the various components of the system during task execution. This is crucial to understand possible system failures before completing recording sessions and gaining real-time insights about model outputs.

[**R2**] **Seamless provenance acquisition**: The future availability of the multimodal dataset produced during a recording session. This supports developers in improving algorithms and debugging system outputs and researchers in retrospectively investigating user-generated data. Therefore, automatically storing the acquired data (and metadata) into databases is important for such a system.

[**R3**] **Retrospective analysis of model performance**: The ability to visualize and inspect large chunks of the acquired data and model outputs to uncover relevant spatial and temporal trends.

[**R4**] **Physical environment representation**: A representation of the physical environment where the performance occurs. This representation should support data exploration tasks by explaining most of the observed user-generated data (e.g. performer movement patterns limited by physical constraints).

[**R5**] **Aggregated and detailed visualization performer behavior**: A summary of the global interaction patterns of the user with the environment. This is key in analyzing general performer behavior. Aggregating large chunks of data temporally and spatially can hide important details, thus, the system should provide both global and local perspectives of performer behavior data.

2.5 ARGUS: Augmented Reality Guidance and User-modeling System

As described in Sec. 2.4, we developed ARGUSconcomitantly with TIM to meet the development needs of building an effective AR task assistant. In total, ARGUSenables the interactive exploration and debugging of all components of the data ecosystem needed to support intelligent task guidance. This ecosystem contains the data captured by the HoloLens's sensors and the outputs of the perception and reasoning models outlined in Sec. 2.3. ARGUShas two operation modes: "Online" (during task performance), and "Offline" (after performance). Users can use these two modes separately if needed, for instance, to perform real-time debugging through the online mode. In another usage scenario, users may start by using the online mode to record a session and then explore and analyze the data in detail using the offline mode. We describe additional usage scenarios in Section 2.6 through two case studies.

2.5.1 ARGUSOnline: Real-time Debugger

Real-Time Debugging The ARGUSarchitecture allows streaming data collection and processing in real-time, which makes instantaneous debugging and data validation possible [**R1**]. As depicted in Fig. 2.2, the online mode provides information on the outputs of the reasoning and perception models using custom visual widgets. The caption of Fig. 2.2 describes each component.

Since what the HoloLens main camera sees (and thus what is analyzed by the models) is not the same as what the performer sees (due to different fields of view), having a real-time viewer such as (A) can help ensure the HoloLens is capturing what the performer and user wish to capture. Additionally, components (C) & (F) provide information that can help validate the objects and actions identified by the models in real-time (as opposed to having to do so post hoc). We note that these features are primarily intended to aid a user in analyzing performer behavior and model performance in real-time, rather than to assist the performer as they complete a task.

Data collection Users of ARGUScan decide when to save the recording for future analysis. By clicking the *Start Recording* button, all data captured and generated by the sensors and models from that point on are redirected from the online streaming database to the historical database until the user clicks *Stop Recording*. The data migration process is transparent to the user [**R2**].

2.5.2 ARGUSOffline: Visualizing Historical Data

The offline mode's main goal is to enable analysis of historical data generated by the models and performer actions in the physical environment [**R3**]. To allow for easy exploration of this large and heterogeneous data, ARGUSprovides a visual user interface that enables querying, filtering, and exploration of the data. Due to the spatiotemporal characteristics of the data, we provide both spatial and temporal visualization widgets to allow users to analyze the data from different perspectives. Fig. 2.3 shows the components composing ARGUSin offline mode. In the following, we describe the main components of the offline mode: the Data Manager, the Temporal View, and the Spatial View. We highlight the interaction flow a user is likely to follow, and for each component, we describe the visualizations, the interactions provided, and their goals.

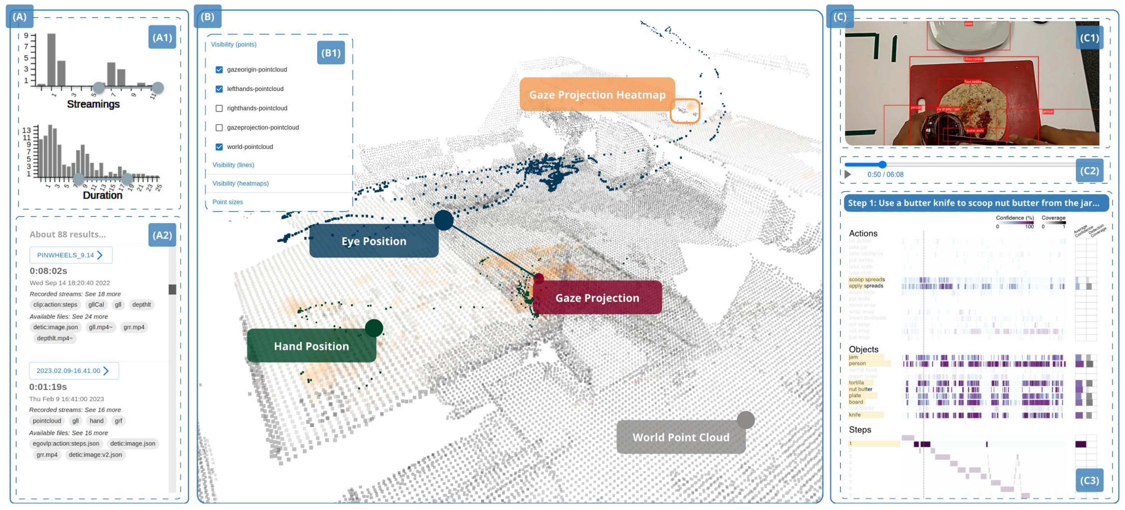


Figure 2.3: Overview of the user interface and components of ARGUSOffline. (A) The **Data Manager** shows the applied filters (A1) and the list of retrieved sessions (A2). (B) The **Spatial View** shows the world point cloud representing the physical environment, 3D points for eye and hand positions, and gaze projections and heatmaps. (B1) Render Controls allow the user to select the elements of the Spatial View they desire to see (C) **Temporal View:** (C1) The Video Player is the main camera video output of the current timestamp selected by the user. (C2) The Temporal Controller controls the video player and updates the model output viewer as well. (C3) The Model Output Viewer displays the output of the machine learning models (reasoning and perception) used during execution time.

2.5.2.1 Data Manager

Users start the exploration by using the Data Manager shown in Fig. 2.3(A) to filter the set of sessions available in the data store. Our data is organized as sessions (each session contains all recordings, data streams, and model outputs for a performer executing a task). The Data Manager enables data retrieval by allowing users to specify filters and select specific sessions from a list of results.

Data Querying Users can query the data by specifying various filters, as shown in Fig. 2.3(A1). Filters are presented in the form of histograms the users can brush to select the desired range.

Query Results The results component displays the retrieved sessions in a list format. Fig. 2.3(A2) shows the results for a given query specified by the user. Each element represents a session showing key features, including name, duration, date, recorded streams, and available model outputs. Once an element of the list is selected, the corresponding data will be loaded into the views of the system.

2.5.2.2 Spatial View

As described in Section 2.3.4, the spatial nature of some of the streamed data demands a 3D visualization to allow users to meaningfully explore the data. For this, ARGUSprovides a Spatial View shown in Fig. 2.3(B) that allows users to analyze how performers interact with the physical environment in conjunction with the spatial distribution of model outputs. The Spatial View can help resolve where performers were located, where they were looking during specific task steps, where objects were located in the scene, etc. Below, we describe the elements of the Spatial View and its interaction mechanisms tailored to support the analysis of the spatial data following well-established visualization guidelines [209] to provide both overview and detailed information.

The basis of the Spatial View is a 3D point cloud (or *world point cloud*) as shown in Fig. 2.3(B) representing the physical environment where the performer is operating [**R4**]. This representation helps us interpret different aspects of the space, such as physical constraints imposed by the environmental layout. However, the point clouds generated based on the data acquired by the headset cameras can easily contain millions of points, making it unfeasible to transfer them over the web and render them within most web browsers. To give an overview of the whole task, all point clouds of one recording are merged to obtain a temporal aggregation. Hololens 2 generates approximately one point cloud per half second, which creates redundancy. This redundancy can be removed by creating a union of all point clouds and then downsampling it using voxelization. However, selecting imprecise parameters can lead to a subrepresentation of the physical space, losing important information and, consequently, hindering analysis. Thus, we parameterize the voxel-based downsampling to create voxels at 1cm resolution, providing enough detail for the purposes of our tool. In our experiments, the downsampled point clouds had less than 100,000 points, even in the worst case, leading to reasonable transfer and rendering times. With the world point cloud representing the physical environment, we are able to visualize performer activity in context.

As illustrated in Fig. 2.3, eye position, hand position, and other data streams can be represented as 3D points in the same scene. The blue dots show the eye position of the performer during a session, while the green dots show the hand position. For each collection of 3D points representing a data stream, users can retrieve more detailed information by interacting with the points. For example, if the user hovers their mouse over the points representing the eye position, a line representing the gaze direction will automatically be rendered in the scene, representing what point in space the performer was looking at from their current position at a specific timestamp. This is possible by calculating the intersection of the gaze direction vector with the world point cloud. This gaze information can also be represented as a 3D point cloud to provide a visual summary of the areas the performer was focused on [**R5**]. This interaction also updates the corresponding video frame in Fig. 2.3(C1) and highlights the models' outputs in Fig. 2.3(C3).

Although the point cloud provides a summarization of the spatial distribution of these data streams, this representation fails to convey aggregated statistics of the data, such as the density of points in a given region, which is proportional to the amount of time the performer spent in a given location of the scene. For this purpose, a 3D heatmap is a more suitable visualization. The Spatial View can create 3D heatmaps of each data stream. The heatmap in Fig. 2.3(B) shows the distribution of the gaze data during the session. We leverage the voxel information created during the downsampling to calculate the density of points within voxels. Using an appropriate color scale, we render cells with non-negligible densities to create the 3D heatmap. Every data stream containing spatial information can be incorporated into the Spatial View as 3D point clouds or heatmaps in ARGUS. Information regarding perception and reasoning models is also available in this view. By combining bounding boxes generated by perception models and depth information captured by the headset, we reconstruct the center point of each detected object, helping users understand the spatial distribution of objects. Also, occupancy maps representing the density of objects in different regions can be derived as presented in Fig. 2.8. Moreover, the Spatial View provides a summarization of gaze information by rendering sets of vectors representing gaze directions over time. Users can control the style (e.g., size and opacity of points) and visibility of all data streams, choosing what data should be visible for analysis. Lastly, point clouds can also be filtered based on timestamp ranges, allowing for focused analysis of specific task steps ("Visibility" in Fig. 2.3(B1)).

2.5.2.3 Temporal View

ML models are a core component of an AI assistant system. While the field of ML has seen many recent advancements to support assistive AR applications [17, 72, 188], the need for tools to improve them remains. Model debuggers are powerful tools used to analyze, understand, and improve these models by identifying issues and probing ML response functions and decision boundaries. This helps developers make models more accurate, fair, and secure, promoting trust and enabling understanding, which is highly desirable in intelligent AR assistants. ARGUSprovides a model debugger based on temporal visualizations to debug the ML models used in AI assistant systems [**R5**]. We describe the different temporal components in detail in the following subsections.

Video Player The object detection model not only recognizes all objects in an image but also their positions. To inspect these outputs, ARGUScontains a video player component that identifies the spatial location of detected objects over time, as shown in Fig. 2.3C. This component allows the user to toggle between two views: 1) the raw main camera video stream and 2) a panoramic mosaic view which consists of a sequence of panoramic mosaics generated from this main camera stream. We highlight all detected objects with bounding boxes, which are provided by the object detection model.

The first view of the video player, which displays the raw main camera video stream collected by HoloLens, enables a highly granular level of model debugging. This allows the user to note specific frames where object detection failed or yielded unexpected results. However, the main camera of the HoloLens has a limited field of view. Often objects that the performer sees at a given timestep cannot be seen in the frame of the raw main camera video at that timestep. Therefore, we aggregate frames into a series of panoramic mosaics in the second view of the video player component, capturing a broader scope of what the performer sees at each timestep. We generate these panoramic mosaics by sampling frames from a temporal window centered around the current timestep. We then compute SIFT features for each frame [134], match them using a Fast Library for Approximate Nearest Neighbors (FLANN)-based matcher [158], and filter for valid matches by Lowe's ratio test [134] before warping and compositing the frames into a panoramic mosaic. We observe that these panoramic mosaics expand the view of the scene significantly, revealing objects within the field of view of the performer at a given timestep that were not captured by the main camera at that same timestep (see Fig. 2.4).

We note that in much of the existing literature on panoramic mosaics, the goal is to capture a seamless wider view of an (often static) scene at a single point in time. In these cases, previous works have endeavored to work around both in-scene and camera motion by excluding moving objects within the scene [103] or only addressing simple, slow camera panning motions [217].

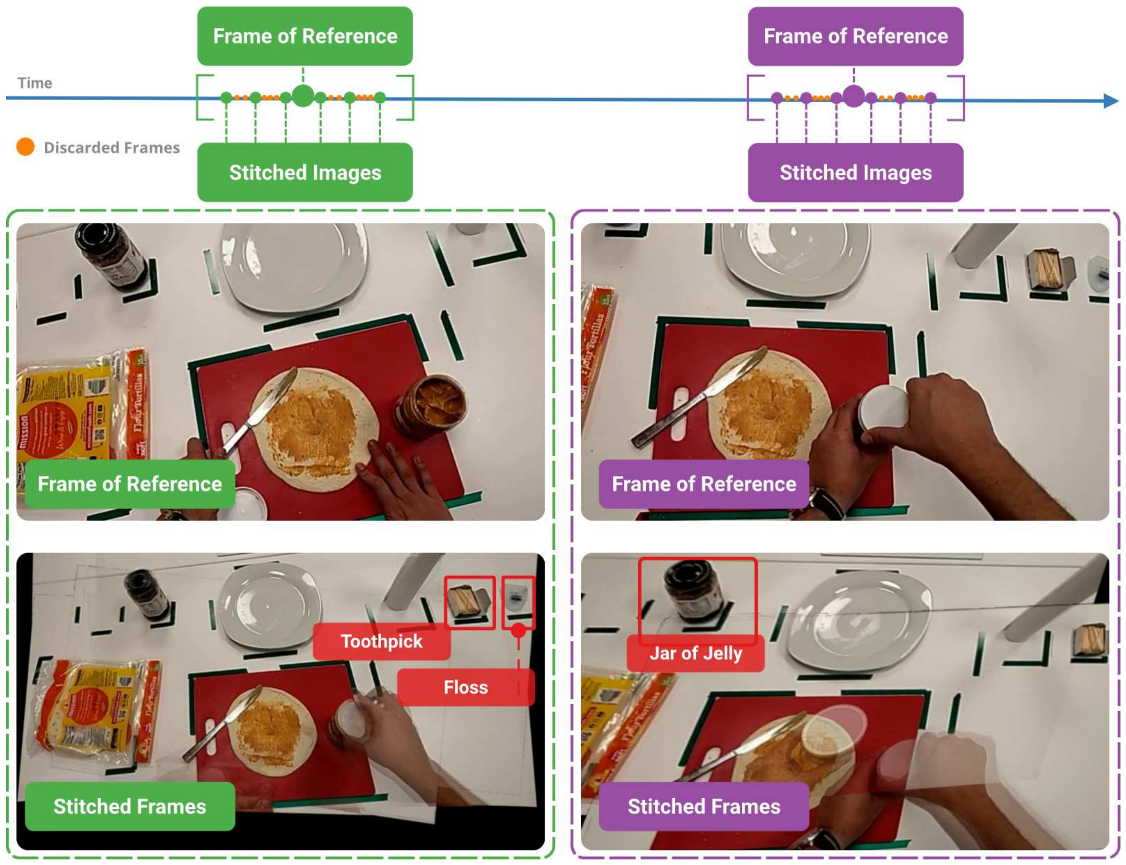


Figure 2.4: A visual representation of frame selection for the panoramic mosaic view of the video player (top) and comparison of these panoramic mosaics with corresponding frames from the same timestep of the raw main camera video for reference (bottom). Each panoramic mosaic is composed of several frames sampled from a window around the current timestep of the raw main camera video. In both examples, we highlight objects that are visible in the panoramic mosaic but not in the raw main camera video (toothpick, floss, and jar of jelly, respectively) in red.

When capturing video from an AR headset of a performer completing a task, however, unpredictable and rapid motion is not only unavoidable, but a valuable indicator of performer behavior. Therefore, our goal extends beyond the typical spatial expansion provided by a panoramic mosaic; we also aim to show how objects move around the complete scene over time, and how the object detection model performs over the given time range in order to facilitate both temporal and spatial analysis of a scene. We note that in our example task shown in Fig. 2.4 (cooking), the performer will often remain in the same

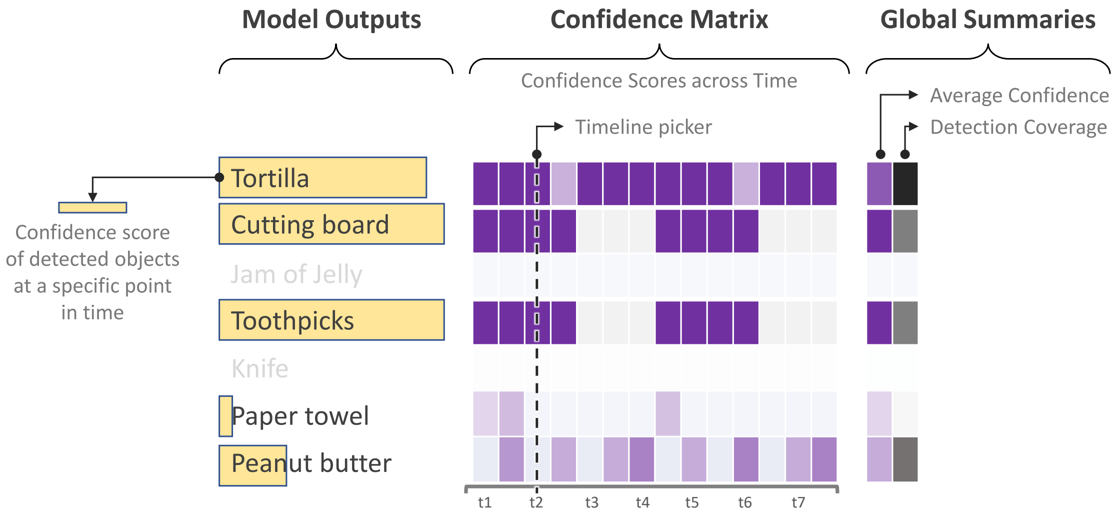


Figure 2.5: Illustration of the Model Output Viewer applied to the analysis of a cooking recipe session. To the left, the model outputs are listed vertically. The bars depict the confidence score of the detected outputs' labels at the specific time picked on the timeline. In the middle, the temporal distribution of ML model output confidences across the whole session is displayed. To the right, two summaries are shown: the average confidence and detection coverage for each output across the entire session. Color darkness is proportional to confidence value: ▨.

position for many consecutive timesteps, consequently, the panoramic mosaic may not significantly expand the field of view at every timestep. Nevertheless, for tasks where the performer traverses a larger area or turns their head in a wider range (and at timesteps where that behavior occurs in this task), the panoramic mosaic will significantly increase the portion of the scene shown at a given timestep.

Model Output Viewer During the debugging process of AR assistant models, the need for model output summaries is key to starting an analysis or evaluation. However, the temporal aspect inherent to these kinds of models makes this task more challenging since they often need to manage the sequence of actions or events chronologically. The Model Output Viewer provides a summarization of the temporal distribution of the ML models outputs across the whole session (see Fig. 2.3(C3)). This visualization is especially useful to find salient patterns, such as quick transitions between steps in step detection models, or to evaluate prediction consistency across time, allowing users to quickly have a global picture of the model behavior, something that could not be achieved by analyzing specific time frames.

As mentioned in Section [**R5**], for AR assistive systems, the most relevant model outputs are the objects, actions, and steps. Once these model outputs are available, they are used to create the matrix visual representations for temporal model analysis. Fig. 2.5 illustrates the Model Output Viewer, where three main components are highlighted: the model outputs, the confidence matrix, and the global summaries. The *Model output* view presents all the model outputs grouped by category. For example, as shown in Fig. 2.3(C3), there are three categories listed vertically: Objects, Actions, and Steps. The object, action, and step sections have multiple rows, each of which lists the model outputs for each category, e.g., the detected objects identified by the perception model. *Confidence matrix*: The x-axis (or columns) indicates the time, from 0 to the total duration of the session (video). Each cell of the matrix is colored according to the confidence score of the detected item at time t (0% ▨ 100%). If no action, object, or step is present, the matrix cell is left blank (white). The total number of cells is proportional to the size of the session (seconds), and all cells are equal in height. Users can hover over the cells to see additional details. *Global summaries*: The Model Output Viewer also provides summaries of the average confidence and detection coverage for each row on the right side of the view so users can quickly evaluate them. The average confidence only takes the confidence value of detected objects, actions, or steps into account. Detection coverage refers to the total number of detections available for each model output (objects, actions, and steps).

Even though the Temporal View representation can provide a visual summary of the temporal distribution of the model output, details-on-demand functionalities remain crucial for debugging. The Model Output Viewer allows users to do a focused analysis by letting them explore the model output results at specific points in time for further analysis. The user can use the temporal controller or the 3D viewer to make this selection. After this, all the objects, actions, and steps detected for that specific point in time that meet the confidence threshold are highlighted, as shown in Fig. 2.3(C3). Users can adjust the confidence threshold value using the slider to investigate the

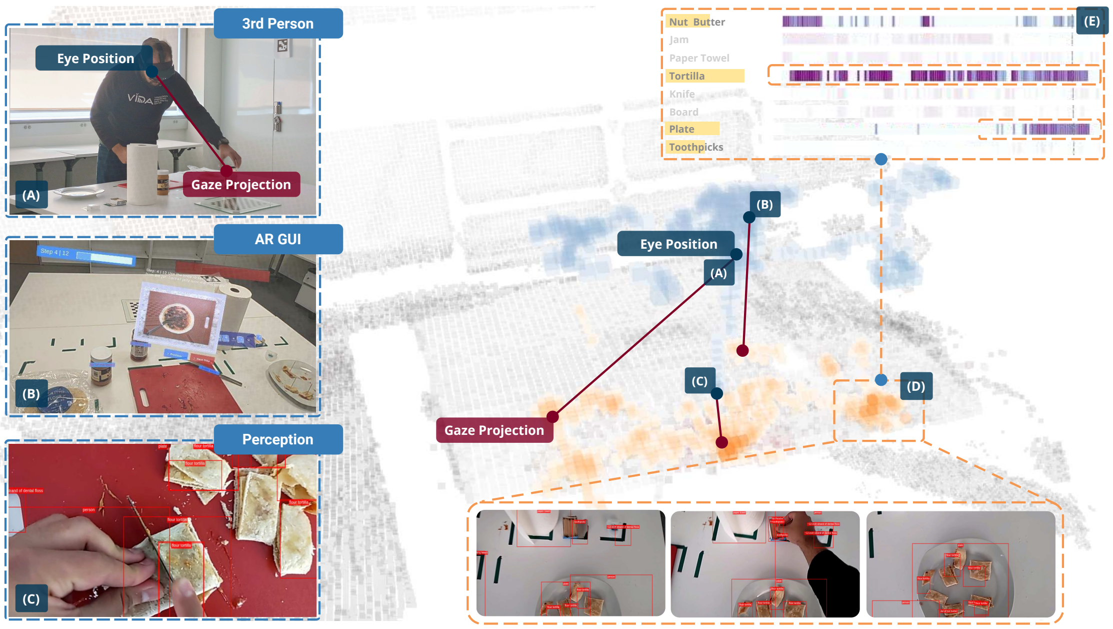


Figure 2.6: ARGUSis a visual analytics tool for real-time and historical evaluation of sensor and model outputs of AR task assistants. Shown here are (A) 3rd person perspective of a human (*performer*) performing a task with AR headset guidance, (B) the AR GUI from the perspective of the performer, (C) a snapshot of a visual perception model analyzing data from the headset camera. For each time step, detailed information highlights the performer's gaze direction and the corresponding frame recorded by the headset. (D) heatmaps of the performer's gaze projection onto the world point cloud allow for inspection and understanding of their attention over time. The cluster on the left shows the performer finalizing a recipe. (E) object detection information from ARGUSTemporal View illustrating that *Plate* was only detected towards the end of the task while *Tortilla* was detected throughout the whole session.

object detection results. We also display object and action labels with bars depicting the confidence value for each label following guidelines of Felix et al. [70] (see Fig. 2.6).

2.5.3 Implementation and Performance Details

The implementation of ARGUSfollows a set of constraints to allow for interactive query and rendering times. The backend supporting the rest API was written using Python and FastAPI [196]. The ML models were trained and/or fine-tuned using PyTorch and serve predictions in real-time utilizing the same streaming protocol used by ARGUS. The interface was structured as a dashboard-like single-page application built with React [216] and TypeScript [145]. The visualization of 3D components uses Three.js [229] and D3 [25]. All the data consumed by ARGUSonline mode comes from querying our Redis database, while the data available in the offline mode comes from the data store in JSON format. All the code is open source and hosted on GitHub [10].

We have measured the latency of Microsoft's Windows Device Portal (part of their mixed reality capture [146]) at ∼1.3 s for streaming the main Hololens 2 camera, while ARGUS has a lower latency of ∼300 ms. Currently, during online use, we save the various data streams at they get off the device. For the session in Section 2.6.2, which takes 1:42 min, the streamed point cloud has more than 10 M points, and it is highly redundant, since the same geometry is sampled over and over again. After the performer finishes a recording, we merge and downsample this data into a consolidated point cloud (see Sec. 2.5.2.2), in this case with 70,000 points. We also create a voxel grid to generate the heat maps, which take 2.3 s. After loading, all data is rendered in real-time.

2.6 Case Studies & Discussion

In this section, we present two case studies describing how model developers have made use of some of the available features. The section ends with feedback from domain experts who have used ARGUSwhile developing AR task guidance software and a discussion of limitations.

2.6.1 Improving Step Transitions in Reasoning Module

To showcase how the Model Output Viewer supports the exploration and analysis of AI assistant model outputs (objects, actions, and steps), we describe how an ML engineer used this tool, the insights they gained, and how the reasoning module of the AI assistant, TIM, was improved through these insights. The ML engineer began by exploring the outputs of the reasoning and perception modules of a recorded session where a performer used TIM to follow a recipe [157].

Analyzing step transitions The visualization of the entire cooking session can help users find salient patterns, e.g., how the transitions between steps were carried out. The first repeated pattern identified by the ML engineer while using ARGUS's Model Output Viewer was the slow transition between steps (see Fig. 2.7). Investigating the "Steps" reveals that steps 1, 3, and 4 were performed over unexpectedly longer periods of time than steps 0 and 2. Also, the user noticed that the model only identified 5 out of 12 steps. Clearly, these two observations indicate that the reasoning module is making errors in identifying recipe steps. This visual summary of the Model Output Viewer allows developers to quickly possess a global picture of model performance and assess errors. This could not be achieved as easily without ARGUS.

Exploring detected objects Under "Objects" in the Model Output Viewer (Fig. 2.7), the ML engineer noticed that the Detic model identified most of the objects for the entirety of a recipe video. This is apparent from the confidence matrix, where most rows are colored (meaning an object was detected). The user also analyzed the confidence values of each detected object. For instance, at the 0:14 mark of the video, objects like *board*, *nut butter*, and *knife* had

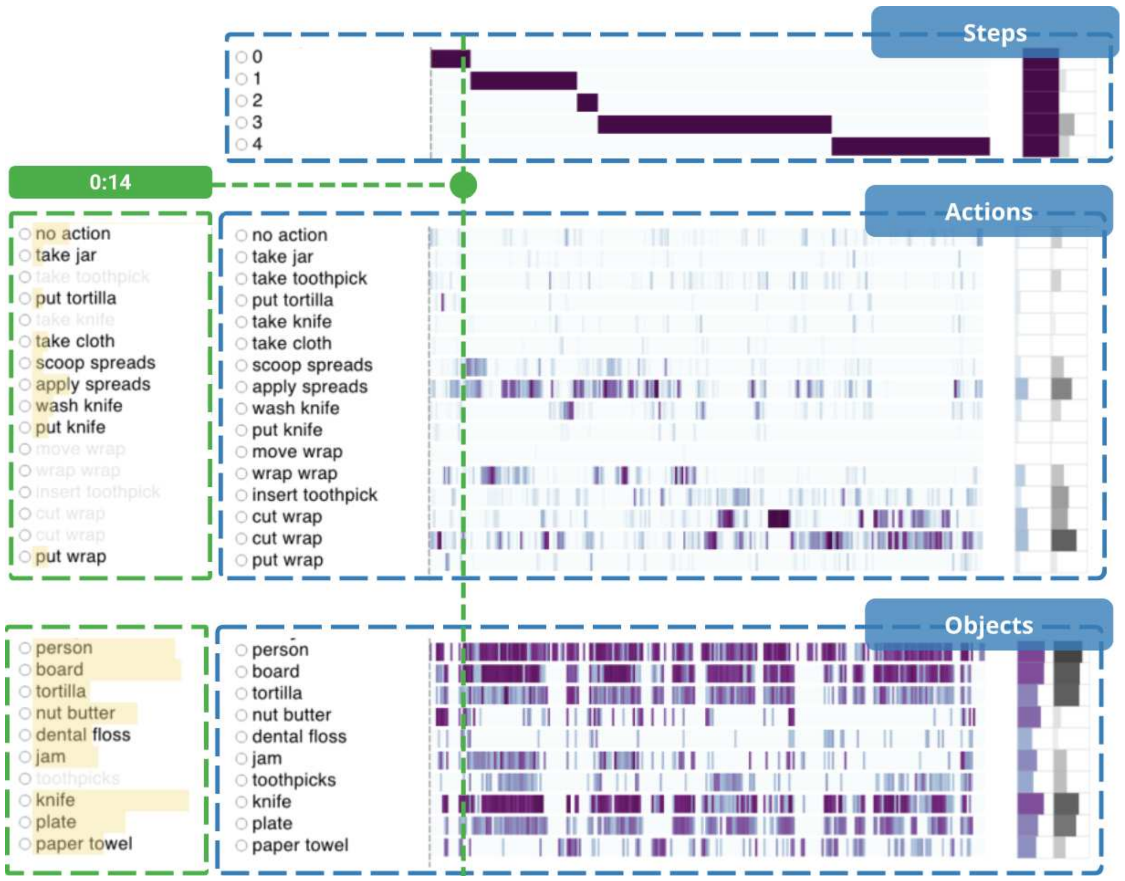


Figure 2.7: Analysis of actions, objects, and steps in the Model Output Viewer. Color darkness is proportional to confidence value (0% ▬ 100%). The confidence matrix and the average confidence views show that the confidence scores for objects are higher than actions. The arrows show the confidence scores for actions and objects at minute 0:14 of the video. The detection coverage view shows that some actions (e.g., *take jar*) are rarely identified during the video.

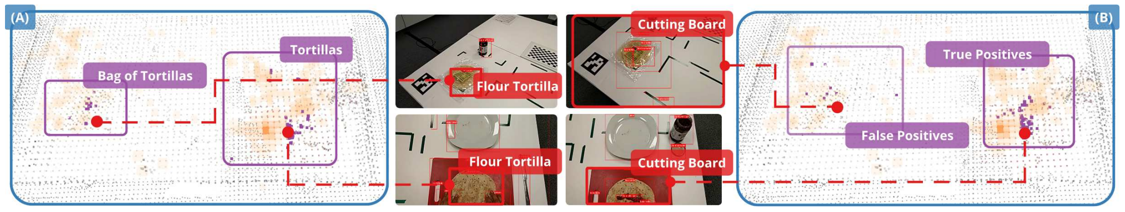


Figure 2.8: Example of how the Spatial View can help users identify missing classes in the model's vocabulary or find clusters of false positives. (A) Points representing the 3D positions the object detection model identified as a "tortilla" during the session. Points on the left represent a bag of tortillas, while points on the right represent a single tortilla. (B) Points representing the 3D positions the object detection model identified as a "cutting board" during the session. The cluster on the left contains false positives, where the perception model generates wrong bounding boxes.

high confidence values, indicated by the yellow background (see zoomed-in views in Fig. 2.7). They could also see this trend in the "Average Confidence" column, which provides an average of confidence values throughout the video.

Identifying missing actions The ML engineer also analyzed the "Actions" section of the Model Output Viewer. They noticed that some actions were rarely detected by the EgoVLP model. As we can see in the "Detection Coverage" column, actions like *put knife*, *move wrap*, and *take cloth* were detected an unusually few number of times. This indicated that it was difficult for EgoVLP to detect those actions. They also noticed that the confidence values for the actions were much lower than the ones for objects. As is visible in Fig. 2.7, the predominant color during the whole session was light purple, which represents low confidence in detecting the actions. Also, at time 0:14 of the video, the confidence values for "scoop spreads" and "apply spreads" were low. In the "Average Confidence" column, we can see that actions such as "wash knife" and "insert toothpick" had approximately 30% confidence. This information led the ML engineer to hypothesize that a decrease in the confidence threshold might be necessary to recognize steps effectively. The visualizations provided by ARGUSalso help to investigate whether lowering the threshold would lead to false positives in the step recognition.

Using insights to improve the reasoning module After the analysis,

Table 2.2: Accuracy of the old and new version of the reasoning module for recognizing the steps of the recipe.

Version	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	Total
Old	1.0	0.9	0.3	0.9	0	0	0	0	0	0	0	0	0.35
New	1.0	1.0	0.5	1.0	0	0	0.4	0.9	0	1.0	1.0	0	0.73

the ML engineer modified the reasoning module to handle actions with low confidence. The reasoning module defaults to selecting actions with greater than 70% confidence. The ML engineer used the confidence slider of Model Output Viewer to tune this value. The most promising value they found was 30%. They reran the new version of the reasoning module for the same video. As shown in Table 2.2, the step estimation accuracy increased for every step and from 35% to 73% overall. Also, this new version was able to identify 8 out of 12 steps, while the previous version only identified 4 out of 12 steps.

Interpreting the new results The Model Output Viewer was also useful for the ML engineer to understand why the reasoning module failed to recognize some steps. As we can see in Table 2.2, steps 4, 5, and 8 were not recognized. For instance, step 5 ("Roll the tortilla from one end to the other into a log shape") is directly related to the action "move wrap", and this action was not identified at all during the entire session (see "Detection Coverage" column). Since this action is necessary for step 5, it was not identified by the reasoning module.

2.6.2 Using Spatial Features to Explain Failures

Although the Temporal View can help users uncover undesired patterns in model performance, it does not paint the full picture of the situation, as model failures might be related to spatial characteristics or performer behavior. In this case study, we show how the Spatial View can provide deeper insights into both reasoning and perception models by assisting users in finding regions where the perception models underperform and to correlate performer behavior with reasoning outputs. A very common way to assess the quality of perception models is by checking the spatial distribution of static objects. In other words, the physical objects captured by the headset cameras can generally be classified as either static objects (objects that will likely not move) or dynamic objects. This classification can help users quickly identify regions where the perception model fails by detecting objects not expected to move throughout a recording session. This case study highlights how this sanity check becomes trivial in ARGUS.

We start the exploration by first using the Data Manager to load the parts of a recording where the perception model underperformed. Once a recording from this period is loaded, we can use the Spatial View to find regions of the space where the performer was interacting. Fig. 2.3 shows points of performer positions (blue) and gaze projection (orange). During an initial inspection, the user can quickly recognize three darker regions projected in the world's point cloud. The rightmost region represents the time during which the performer was interacting with ARGUSin online mode to start the recording, while the other two regions are on the desk. The user then hovers the mouse over the points on the 3D point cloud representing the gaze projection on the world point cloud to look at the corresponding video frames in Temporal View. This interaction reveals that the left region contains the ingredients for the recipe, while the actions (e.g., "spreading jelly on the tortilla") happen on the right side. By highlighting the heatmaps only, it becomes clear that the performer spent most of their time looking at the right side of the table (darker region), meaning the performer spent more time executing actions than selecting ingredients. With this understanding of the spatial distribution of the performer's attention, the user can infer that the model outputs high confidence values for *tortilla* and *cutting board* throughout the entire session as shown in Fig. 2.7, which makes the user question the validity of the output. Then, the user displays the 3D point cloud denoting the 3D positions of tortillas shown in Fig. 2.8(A). Two clusters show up, allowing the user to look at the corresponding frames, realizing that the left cluster represents a bag of tortillas while the other is the tortilla used for the recipe. This process highlights the need for a more comprehensive class vocabulary able to represent both tortillas and bags of tortillas. Following that, the same process is conducted for the *cutting board* class, and a similar pattern with two clusters arises (see Fig. 2.8(B)). Since the cutting board was a static object in the recording, the user can quickly realize that one of the clusters may be representing a model failure. The corresponding video frames selected interactively confirm that the left cluster contains only false negatives. Lastly, inspecting the bounding boxes rendered on the video frames (see Fig. 2.8) gives the user more detailed information about the error. In this case, the user sees that the model is generating bounding boxes covering almost the entire field of view of the performer.

2.6.3 Expert Feedback

The Model Output Viewer provides a visualization of object and action detections with model confidence levels. Even if a model performs very well on an offline evaluation dataset, when deployed in real-time, it will inevitably be presented with previously unseen conditions such as room lighting, skin pigmentation, or object angle. This is known as the “domain-shift” problem [273], where a model fails to perform when presented with data not well represented in its offline evaluation dataset. ARGUSstreamlines real-time deployment, and its Model Output Viewer enables the evaluation of model confidence in a virtually unconstrained domain. This sheds light on which conditions the models perform best, and informs how model robustness could be enhanced by expanding data with new collection or augmentation strategies.

ARGUSis also useful for scenarios where multiple information sources must be analyzed at the same time. For TIM’s reasoning module, which consumes multiple inputs in parallel (e.g., the detected objects/actions and its confidence scores), ARGUS’s visualizations allow the user to understand the reasons the system made the predictions and under what circumstances it succeeded/failed. As shown in Sec. 2.6.1, this tool helped the ML engineer to improve the system.

As ML models develop new capabilities and produce richer representations, it becomes increasingly important to develop scalable visualizations of those outputs. Conventionally, ML engineers either log outputs to the terminal or use drawing libraries to bake the predictions on top of the video. However, there is limited real estate when drawing on a video, and often the predictions and their associated text make it difficult to view the underlying image frames. In contrast, ARGUSprovides a high level of interactivity, which allows it to selectively visualize relevant information while allowing the user to change the view and granularity of this information to suit their needs. Additionally, being able to contextualize and explore ML model outputs in 3D can lead to a better understanding of how model outputs can change based on the perspective, and spatially ground the predictions for an entire recording in a single view. Overall, tools like ARGUSdrastically lighten the visualization load placed on ML engineers and provide a convenient tool for understanding their models.

Limitations While useful for exploration of spatiotemporal data captured by an intelligent assistant, ARGUSneeds more robust data processing algorithms. For instance, in sessions where the performer's hands are recurrently in the field of view of the headset camera, the point cloud generation process captures and transforms it into points of the world space, resulting in potential noise that does not represent the physical environment. To overcome this problem, we review recordings with noisy point clouds and define bounding boxes representing regions where these noisy points must be excluded from the final rendering. We plan to explore methods [65, 147] to automatically remove point cloud noise during run-time acquisition.

2.7 Conclusion & Future Work

We presented ARGUS, an interactive visual analytics system that empowers developers of intelligent assistive AR systems to seamlessly analyze complex datasets created by integrating multiple data streams at different scales, dimensions, and formats acquired during performance time. Furthermore, through interactive and well-integrated spatial and temporal visualization widgets, it allows for retrospective analysis and debugging of historical data generated by ML models for AI assistants.

We envision ARGUSto unlock several avenues for future research connecting human-computer interaction, visualization, and machine learning communities revolving around the goal of developing better and more reliable AR intelligent systems. In the future, we intend to conduct a deeper evaluation of our system's performance metrics (e.g. rendering times, stream latency). We also plan to explore how to extend the system to support the comparison of sessions of multiple performers. This includes the data and model outputs and will require registration of the point clouds. User-generated data acquisition (annotation) and integrated AI techniques during exploration time (segmentation and model training based on the annotated data) are other fronts we would like to cover. Since our Temporal and Spatial Views allow users to explore data and output models across the entire session, adding annotation capabilities is a natural next step. Furthermore, we want to investigate privacy-preserving methods for storing and streaming the collected data, similar to ones that have been proposed, e.g., for eye-tracking data [27, 53], to prevent performer identification.

Chapter 3

StreetAware: A High-Resolution Synchronized Multimodal Urban Scene Dataset

3.1 Introduction

Driven by continuous improvements in computational resources, bandwidth optimization, and latency, activity-rich traffic intersections have been implicated as excellent locations for smart city intelligence nodes [118]. Audio and video sensors located at intersections are, thus, capable of generating large amounts of data. Concomitantly, deep learning and edge computing of these data allow for real-time geospatial mapping and analysis of urban intersection environments, including moving entities such as pedestrians and vehicles. Intersections are some of the most critical areas for both drivers and pedestrians. They are where vehicles and pedestrian paths cross most frequently. Globally, pedestrians represent 23% of the 1.35 million worldwide road traffic deaths every year, with most events occurring at pedestrian crossings [211, 247]. Thus, predicting pedestrian trajectories at intersections and communicating this information to drivers or assisted/autonomous vehicles could help mitigate such accidents. Understanding an intersection scene has significant implications for self-driving vehicles in particular. Figure 3.1 outlines the concept of enhancing the safety of traffic participants by providing real-time insights into out-of-sight events at intersections using a combination of multimodal sensing and edge and in-vehicle computing. In this example, pedestrians and semi-autonomous cars are detected by sight (cameras) and sound (microphones) at intersections, and the information is relayed to each car's self-driving system. In this process, edge computing and cloud communication help inform the intelligent transportation network with real-time information.

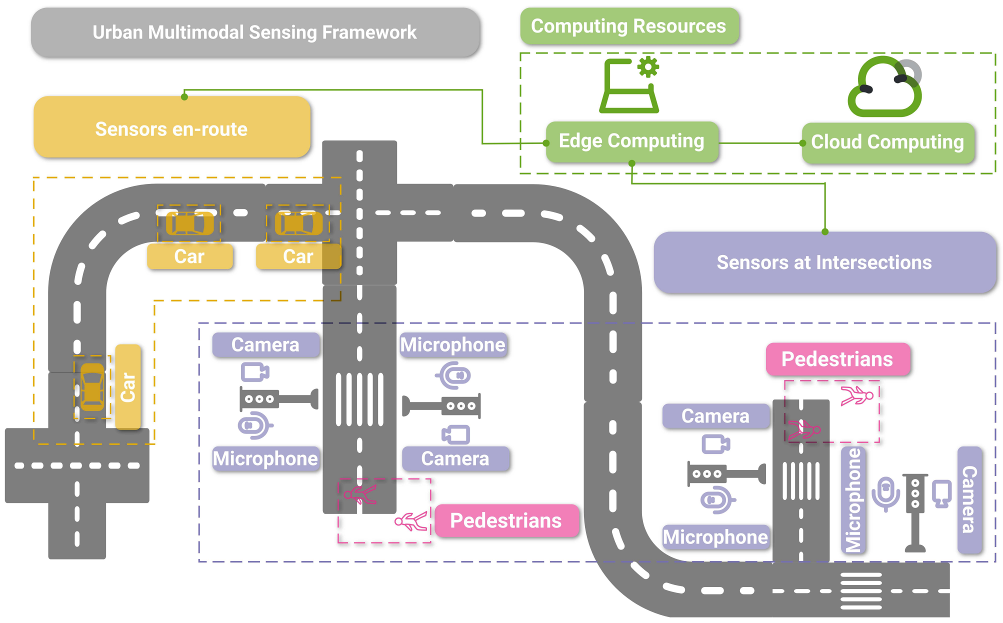


Figure 3.1: Illustration of the basic concept of combining multimodal sensors at critical nodes (e.g., intersections) with on-device and in-vehicle computing capabilities to provide greater awareness to urban traffic participants.

Within the navigation system of an autonomous vehicle, its control system must have detailed, accurate, and reliable information as it approaches such a scene to determine, for instance, the number of road entries into an upcoming crossing or pedestrian and vehicle trajectories to avoid collisions [12]. For such purposes, urban analytical data should have high precision, granularity, and variation (such as multiple perspectives of the same area) to be effectively helpful.

In this work, we present $7\frac{3}{4}$ h of synchronized data collected at urban intersections by specialized Reconfigurable Environmental Intelligence Platform (REIP) sensors developed by the Visualization and Data Analytics (VIDA) Research Center at NYU [186]. REIP sensors are capable of dual 5MP video recording at 15fps, as well as 12-channel audio at 48kHz, for recording pedestrian and vehicle traffic at various locations. We selected three intersections in Brooklyn, New York, with diverse demographic, urban fabric, and built environment profiles and equipped each with four REIP sensors. The sensors were placed at each corner of the intersection and recorded the dynamics of pedestrian and vehicle interaction for several $\approx$40 min sessions, resulting in a total of $\approx$2 TB of raw audiovisual data. The data were synchronized across all sensors with high accuracy for both modalities (one video frame and one audio sample, respectively) using a custom time synchronization solution detailed later. High-synchronization is important so that events that happen across cameras and between video and audio can be viewed and analyzed together with reduced effort, and with confidence, those events actually occurred at the time inscribed in the data.

The presented dataset, which we call StreetAware, is unique to other street-level datasets such as Google Street View because of the following combination of characteristics:

- Multimodal: video, audio, LiDAR;
- Multi-angular: four perspectives;
- High-resolution video: 2592×1944 pixels;
- Synchronization across videos and audio streams;
- Fully anonymized: human faces blurred.

To demonstrate these key features of the data, we present four uses for the data that are not possible on many existing datasets — (1) to track objects using the multiple perspectives of multiple cameras from both audio (sound-based localization) and visual modes, (2) to associate audio events with their respective visual representations using audio and video, (3) to track the amount of each type of object in a scene over time, i.e., occupancy, and (4) to measure the speed of a pedestrian while crossing a street using multiple synchronized views and the high-resolution capability of the cameras.

Our contributions include:

4.1. The StreetAware dataset, which contains multiple data modalities and multiple synchronized high-resolution video viewpoints in a single dataset;

4.2. A new method to synchronize high-sample-rate audio streams;

4.3. A demonstration of use cases that would not be possible without the combination of features contained in the dataset;

4.4. A description of real-world implementation and limitations of REIP sensors.

The data presented here will enable other researchers to develop unique applications of machine learning to urban street-level data, such as modeling pedestrian-vehicle interactions and recognizing pedestrian attributes. Such analysis can subsequently help inform policy and design decisions made in the context of urban sensing and smart cities, including accessibility-aware design and initiatives like Vision Zero. Among the other possibilities we discuss later, further analysis of our data can also shed light on the optimal configuration needed to record and analyze street-level urban data.

This chapter is structured as follows. In Section 3.2, we review some of the literature on street-view datasets and how these types of data have been analyzed with ML-based techniques. In Section 3.3, we discuss our custom sensors and detail data acquisition and processing, with emphasis on the precise synchronization of multiple data modalities (i.e., audio and video). We lay out the motivation for and demonstrate the potential applications of the data in Section 3.4 and provide a discussion and concluding remarks in Sections 3.5.

3.2 Related Work

In this section, we will review some of the currently available audiovisual urban street-level datasets, then succinctly review applications of such data related to deep learning-based object detection, pedestrian tracking, and safety.

3.2.1 Datasets

A handful of related datasets exist. The first is the popular Google Street View [89]. Released in 2007, at a time when a limited number of cities had their own street-level photography programs, Google Street View was revolutionary in that it combined street-level photography with navigation technology. Publicly available but not entirely free, Google Maps Street View includes an API and extensive street-level image coverage throughout much of the World's roadways. Unlike StreetAware, Google Street View is a collection of disparate images instead of stationary video recordings of specific places. Moreover, Google Street View often has multiple viewpoints that are in close proximity to one another, but they do not overlap in time. Therefore, synchronization across multiple views is not possible. Another dataset is Mapillary [242]. Mapillary street-level sequences contain more than 1.6 million vehicle-mounted camera images from 30 major cities across six continents, distinct cameras, and different viewpoints and capture times, spanning all seasons over a nine-year period. All images are geolocated with GPS and compass, and feature high-level attributes such as road type. Again, these data are not video or synchronized and do not include audio. The next dataset is Urban Mosaic [153], which is a tool for exploring the urban environment through a spatially and temporally dense data set of 7.7 million street-level images of New York City captured over the period of one year. Similarly, these data are image-only and unsynchronized across views. Another street-level urban data set is SONYC [36]. SONYC consists of 150 million audio recording samples from the "Sounds of New York City" (SONYC) acoustic sensor network and is aimed at the development and evaluation of machine listening systems for spatiotemporal urban noise monitoring. However, SONYC does not contain visual information. Finally, there is Urban Sound & Sight (Urbansas) [80], which consists of 12 h of unlabeled audio and video data along with 3 h of manually annotated data, but does not contain multiple views. These and other street-level datasets (most oriented toward self-driving vehicle research) are listed in Table 3.2.1 with brief descriptions of each. StreetAware is unique in that it combines stationary, multi-perspective, high-resolution video and audio in a synchronized fashion.

Dataset	Location	Size	Description	Annotations?
GSV [89]	100+ countries	>220 B	Vehicle-mounted camera images; download not free	No
Mapillary [242]	30+ cities	>1.6 M	Vehicle-mounted camera images; condition-diverse; GPS-logged	No
Urban Mosaic [153]	New York	7.7 M	Vehicle-mounted camera images	No
SONYC [18]	New York	150 M	10-second audio samples	Yes
Urbansas [80]	Europe	15 h	10-second audio & video samples	Yes
KITTI [84]	Germany	1 k	Vehicle-mounted camera images; laser scans; GPS-logged	Yes
NuScenes [34]	Boston, MA	1.4 M	Vehicle-mounted camera images; radar & LiDAR; multi-camera	Yes
Waymo [223]	USA	1 M	Vehicle-mounted camera images; LiDAR; condition-diverse	Yes
I2V-MVPD [19]	Tunisia	9.48 k	Vehicle-mounted & stationary synchronized images	Yes
EuroCity Persons [28]	30+ cities	47 k	Vehicle-mounted camera images; condition-diverse; pedestrian-oriented	Yes
PIE [199]	Toronto	911 k	Vehicle-mounted camera images; pedestrian & vehicle-oriented	Yes
KrishnaCam [213]	Pittsburgh, PA	7.6 M	Images from Google Glasses on pedestrian	No
MEVA [49]	Indiana	9.3 kh	Stationary RGBIR & UAV video	Yes
Neovision2 Tower [40]	California	20 k	Stationary camera images	Yes
Cityscapes [48]	50+ cities	25 k	Vehicle-mounted camera images	Yes
NightOwls [175]	Europe	279 k	Vehicle-mounted camera images at night	Yes
Cerema [51]	Controlled environment	62 k	Stationary camera images of pedestrians; varied rain/fog/light conditions	Yes
StreetAware	Brooklyn, NY	7.75 h	Stationary audio & video; synchronized, multi-perspective	No

Table 3.1: Summary of available street-level datasets

3.2.2 Deep Learning Applications

A number of recent studies have explored the use of deep learning for detecting and analyzing objects in street-level audio and video data. A study by Zhang et al. [261] developed an approach to automatically detect road objects and place them in their correct geolocations from street-level images, relying on two convolutional neural networks to segment and classify. Doiron et al. [61] showed the potential for computer vision and street-level imagery to help researchers study patterns of active transportation and other health-related behaviors and exposures. Using 1.15 million Google Street View (GSV) images in seven Canadian cities, the authors applied PSPnet [267], and YOLOv3 [200] to extract data on people, bicycles, buildings, sidewalks, open sky, and vegetation to create associations between urban features and walk-to-work rates. Charitidis et al. released a paper in 2023 [41] in which they utilized several state-of-the-art computer vision approaches, including Cascade R-CNN [35] and RetinaFace [58] architectures for object detection, the ByteTrack method [265] for object tracking, DNET architecture [252] for depth estimation, and DeepLabv3+ architecture [43] for semantic segmentation to detect and geotag urban features from visual data. Object detection systems have also been specifically developed for the collection and analysis of street-level imagery in real-time [219]. In "Smart City Intersections: Intelligence Nodes for Future Metropolises" [118], Kostec et al. detail intersections as intelligence nodes using high-bandwidth, low-latency services for monitoring pedestrians and cloud-connected vehicles in real-time. Other computer vision applications to urban street view imagery include extracting visual features to create soundscape maps [268], mapping trees along urban street networks [135], estimating pedestrian density [230] and volume [42], associating sounds with their respective objects in video [80], and geolocating objects from a combination of street-level and overhead imagery [170].

Pedestrian speed and trajectory prediction are some of the primary computer vision goals in the urban data analytical community, especially in the field of advanced driver assistance systems [211]. The performance of state-of-the-art pedestrian behavior modeling benefits from recent advancements in sensors and the growing availability of large amounts of data (e.g., StreetAware) [117]. A study by Kuo et al. [231] compared estimations of pedestrian speed from a classical model and a neural network in corridor and bottleneck experiments, with results showing that the neural network can better differentiate the two geometries and more accurately estimate pedestrian speed. Ahmed et al. [5] sought to use a fast region-convolutional neural network (Fast R-CNN) [88], a Faster R-CNN [201], and a Single Shot Detector (SSD) [130] for pedestrian and cyclist detection based on the idea that automated tracking, motion modeling, and pose estimation of pedestrians can allow for a successful and accurate method of intent estimation for autonomous vehicles. Other related studies in the literature include applying deep learning techniques for the prediction of pedestrian behavior on crossings with countdown signal timers [78], mapping road safety from street view imagery using an R-CNN [203], and identifying hazard scenarios of non-motorized transportation users through deep learning and street view images in Nanjing, China [239].

3.3 The StreetAware Dataset

In this section, we will first review existing audiovisual sensor options and make the case for harnessing the REIP sensors (Figure 3.2), which were custom-designed and constructed at our lab. Next, we describe how the data are collected, processed, and synchronized.

3.3.1 REIP Sensors

A multi-view requirement for our data collection could be easily satisfied with off-the-shelf video surveillance systems, which often include a set of wireless IP cameras. These cameras transmit their video feeds to a central data storage location, typically a local hard drive, which can sometimes be synchronized with the cloud but is not required for the system to operate. The cameras may also include a night mode, which can prove beneficial during low-light conditions. However, these cameras rarely provide audio due to privacy concerns and rely on manually configured timing information or NTP (Network Time Protocol) for timestamping the video. The latter is a significant barrier to a multi-view analysis of fast-moving objects such as cars. A car traveling at 40 mph covers more than a meter of ground per frame when recorded at 15 fps. Therefore, frame-accurate video synchronization is also a requirement for our dataset, and unfortunately, it cannot be met with off-the-shelf security cameras, which often operate at reduced frame rates due to limited storage.

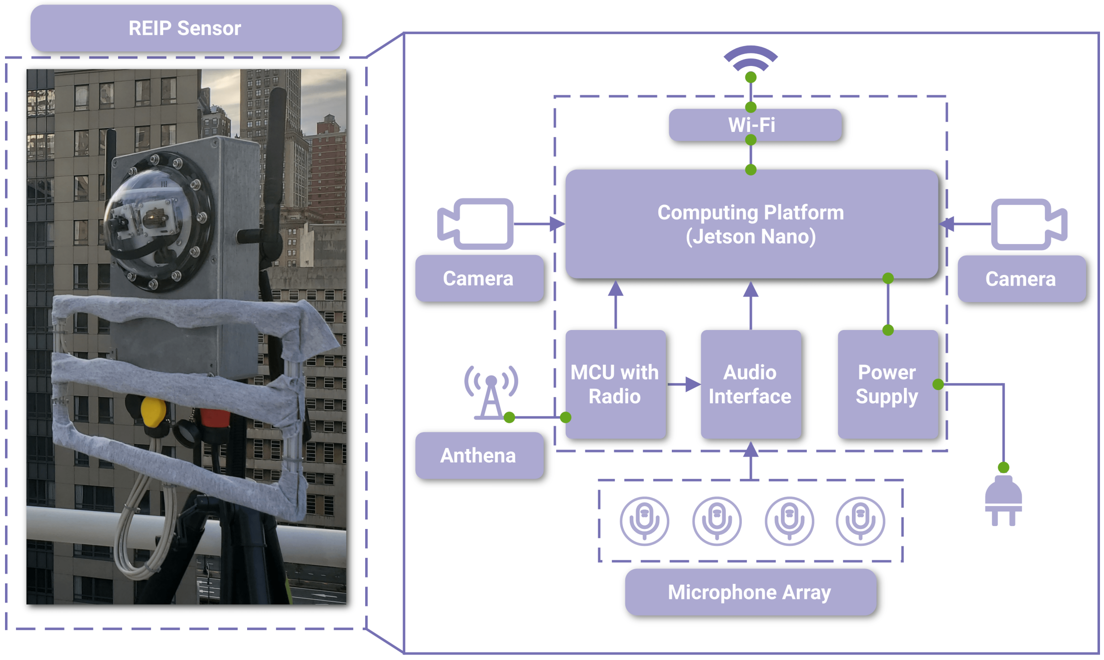


Figure 3.2: A photo of the REIP sensor in its protective metal housing ready for deployment (**left**) and its internal architecture (**right**).

There exist commercial motion tracking systems that use high-speed cameras synchronized by NTP. Although these cameras provide high temporal resolution and accuracy for video, they are insufficient for synchronizing audio data, which requires sub-millisecond timing accuracy. Furthermore, such cameras are typically designed for indoor infrared light imaging, are costly, and rely on an Ethernet interface for synchronization and data transfer, which makes them impractical for larger-scale urban deployments.

Another commercial device that provides quality video with audio at a reasonable price is the GoPro camera. However, the GoPro was designed for independent operation, so it does not feature quality synchronization across multiple cameras. Moreover, synchronization across video and audio modalities is also known to be a problem due to audio lag offset and differences in sampling frequencies. Recently, GPS-based time-coding has been introduced in the latest versions of GoPro cameras. This could help with synchronizing the start of the recordings, but does not solve the ultimate problem of long-term synchronization. The time drift caused by manufacturing variations of the internal crystal oscillator's frequency that drives digital logic (including the sampling frequency) is also susceptible to temperature-based variations. Moreover, there is no way to know when the GoPro is experiencing lost frames during recording, which ruins the single timestamp-based synchronization altogether. The solution would be a continuous (re)synchronization of the cameras from a single clock source during the entire recording process. Other potential issues include remote control and monitoring of the camera's status, as well as weatherproofing that may require external devices and housing, depending on the camera version.

Ultimately, the sensors used in this study are custom-built in our lab. An overview of the sensor's architecture is provided in Figure 3.2 with its specifications listed in Table 3.3.1. The sensor includes two cameras and a microphone array. It also features a high-precision custom synchronization solution for both video and audio data based on a 2.4 GHz radio module receiving common global timestamps from a master radio device. Each camera records 5 MP video at 15 fps, and the microphone array records audio through 12 synchronized channels. The custom acoustic front-end was designed to capture audio from the 4×3 digital pulse density modulated (PDM) micro-electro-mechanical systems (MEMS) microphones. It uses the USB MCHStreamer as an audio interface, which is a USB audio class-compliant device, making it compatible with the readily available microphone block in the REIP SDK [186]. Each sensor has 250 GB of internal storage and is operated on a FlashFish portable power station. The computing core is the NVIDIA Jetson Nano Developer Kit, which offers edge-computing capabilities. The majority of the sensor's hardware is enclosed within the weatherproof aluminum housing. Heat sinks are designed to offer resistance to extreme temperatures, providing better performance. For sensor control, a locally deployed network router and Wi-Fi connectivity are used.

Feature	**Specification**
Internal Storage	250 GB
Power capacity	300 Wh
Camera resolution	5 MP
Camera field-of-view	160° (85° max per camera)
Camera frame rate	15 fps (nominal)
Audio channels	12 (4×3 array)
Audio sampling rate	48 kHz
NVIDIA Jetson Nano GPU and CPU cores	128 and 4
NVIDIA Jetson Nano CPU processor speed	1.43 GHz
NVIDIA Jetson Nano RAM	4 GB LPDDR4

Table 3.2: REIP sensor specifications including its two cameras, 12-channel microphone array, and NVIDIA Jetson Nano as a computing platform.

REIP sensors provide high-resolution video and audio recording with an inbuilt synchronization solution (the high-level architecture is shown in Figure 3.2). Both cameras and audio interface are USB 2.0 devices. Of note is the design of the audio pipeline where the MCHStreamer interface is receiving an additional audio-like signal from the microcontroller unit (MCU). The purpose of this signal is to embed the global timing information received by the radio module as additional audio channels. For video, the individual image frames are timestamped by the NVIDIA Jetson Nano as they arrive into the camera block of the data acquisition and processing pipeline powered by REIP SDK. For that, the MCU is also connected to the computing platform via a USB 1.1 interface and continuously provides the latest global timestamp transmitted to each sensor by a master radio module (a separate device).

3.3.2 Data Collection

Three intersection locations were selected to acquire the dataset with different road configurations and pedestrian demographics as described below:

4.1. *Commodore Barry Park.* This intersection is adjacent to a public school. It has a low-to-medium frequency of traffic, making it an uncrowded intersection.

4.2. *Chase Center.* This intersection is adjacent to the Chase Bank office building within Brooklyn's MetroTech Center. It is also an active pedestrian intersection.

4.3. *DUMBO.* The intersection of Old Fulton Street and Front Street is under the Brooklyn Bridge. Being a tourist destination, this intersection is the busiest of the three. Due to smaller crosswalks and heavy traffic, it presents challenges such as occlusion and a diverse range of pedestrian types.

Overhead map locations and the sensors' positions for the recording sessions at Commodore Barry Park as an example are shown in Figure 3.3. Each sensor is equipped with two 5 MP USB cameras providing a combined 160° horizontal field of view at a recording rate of 15 fps. The 4 × 3 microphone array of each sensor records at a sampling rate of 48 kHz. Every sensor was powered by a portable power station with a 300 Wh capacity. An Ouster OS-1 LiDAR sensor is also included. It has a configuration of 16 vertical scanning lines at 1° angular resolution and 1024 samples per revolution.

We used four REIP sensors at each intersection, one placed at each corner of the intersection. We recorded several 30–45 min long sessions at each intersection—four at Commodore Barry Park, three at Chase Center, and four at DUMBO. This results in about 200 GB of raw audiovisual data

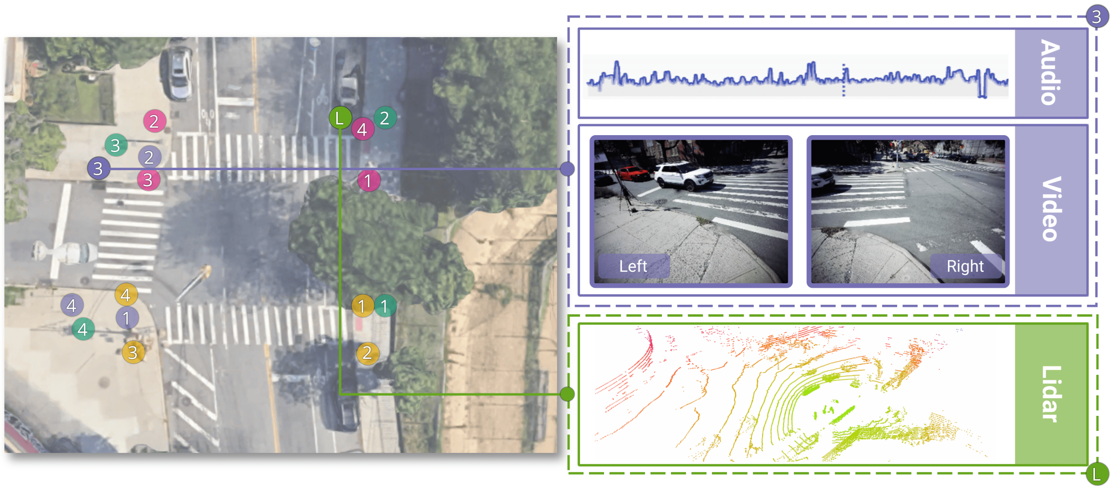


Figure 3.3: Illustration of the sensor positions and data types at the Commodore Barry Park intersection. Colors denote the different recording sessions, and numbers indicate the REIP sensors. This figure highlights all data modalities that are being captured during the collection process: audio, video, and LiDAR scans. Green L indicates the LiDAR sensor position fixed for all recording sessions.

recorded by each sensor per location (limited by the sensor's max storage capacity of 250 GB). In total, we collected ≈2 TB of raw data.

The data acquisition pipeline of the sensors is shown in Figure 3.4. The pipeline is based on the software blocks available in the REIP SDK as released in [186]. Because our sensors are based on the budget NVIDIA Jetson Nano computing platform, a slight modification was necessary for a camera block to be able to timestamp every frame from both cameras for synchronization purposes. We bypass the decoding of JPEG images sent by the cameras to free up CPU resources. Instead, we direct the raw video stream to the file using features of GStreamer library [1] that the camera block is based upon. Still, we did experience some lost frames when recording during the summer month of August due to the throttling of the sensor's NVIDIA Jetson Nano computing platform after prolonged exposure to extreme temperature conditions.

The output of the sensor's data acquisition pipeline contains three types of data: 500 MB chunks of video data (approximately one minute of recording, depending on the intersection), JSON files containing batches of timestamps

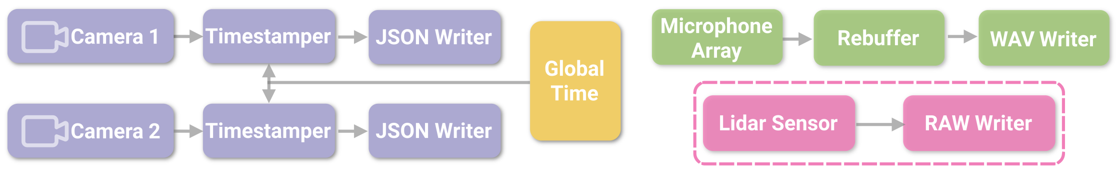


Figure 3.4: The sensor's data acquisition pipeline is built using software blocks available in the REIP SDK. It contains separate tasks for each camera and the microphone array. The LiDAR data acquisition is performed on a separate machine (orchestrator laptop).

for each frame in the video data chunks, and 5-second long chunks of audio data with its timing information embedded as extra audio channels. We spare the users from working with the sensor's raw data by preprocessing it, including anonymization and synchronization. We also use a space-efficient video codec, H.264, instead of the camera's original MJPEG data stream. Table 3.3 summarizes the specifications of the processed dataset that we are releasing.

3.3.3 Data Synchronization

In this section, we detail our synchronization techniques, first for audio, then for video data modality. The synchronization techniques are independent for each modality. Figure 3.5 illustrates the overall principle.

The method is fundamentally reliant on the hardware design of the sensors where the communication delay between the master radio and each sensor's slave radio is constant, and the radio waves propagation delay is negligible due to the large speed of light of 299,792 km/s. Similarly, the data readout latency for the cameras is equal across sensors because of the identical cameras used. The video modality can then be synchronized with audio by calibrating the frame readout latency. For that, a rapid event with a loud sound, such as a clap, is recorded in close proximity to the sensor (for negligible sound propagation delay). The true time of the event is then deducted based on sound and compared to the latest global timestamp received by the computing platform when the video frame is released by the driver into a REIP pipeline.

Table 3.3: Dataset specifications after processing, featuring 3 data modalities (audio, video, and LiDAR) with synchronized footage.

Feature	**Specification**
Number of geographic locations	3
Number of recording sessions	11
Typical recording length	30–45 min
Total unique footage time	465 min (7.75 h)
Total number of image frames	≈403,000
Video resolution	2592 × 1944 pixels
Number of data modalities	3
Synchronized and anonymized	True
Video synchronization tolerance	2 frames
Audio synchronization tolerance	1 sample
Total audio & video size	236 GB
Total LiDAR size	291 GB
Total size	527 GB

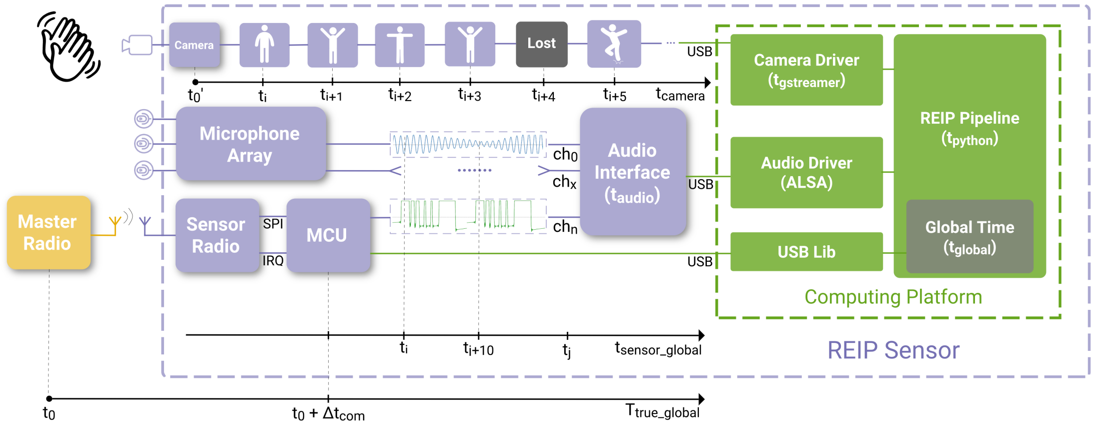


Figure 3.5: Multimodal synchronization workflow. Each sensor is receiving the global timestamps from a master radio (at 1200 Hz) and is embedding every 10th of them in a serialized form as an extra audio track synchronous with the microphone array data. For the cameras used in REIP sensors, it is not possible to embed the timing information directly into the video data itself. Instead, the timestamps provided by the camera driver are converted into the global timeline using the computing platform that is continuously updating the latest timestamp received by the microcontroller unit (MCU) via USB. More details on video synchronization

3.3.3.1 Audio

Audio synchronization is a challenging task because audio data are being sampled at a very high rate, 48 kHz in the case of our sensors. Furthermore, the speed of sound wave propagation in the air is $c = 343$ m/s which translates into a synchronization accuracy requirement of less than one millisecond, across all sensors, for any meaningful audio-based sound source locations to work. Such accuracy cannot be achieved by simply attaching a timestamp to the chunks of audio provided by the driver because of the large 'jitter' of such timestamps caused by the random operating system (OS) interrupts on the computing platform. Therefore, the synchronization information must be embedded into the audio data itself before it even makes it to the audio driver of the OS. In this subsection, we introduce a novel method for high-accuracy audio synchronization by means of embedding a special signal into a dedicated audio channel of the audio interface (Figure 3.6).

The radio module of each sensor receives a global timestamp from a master radio transmitting it at a rate of 1200 Hz. Unlike the operating system of the computing platform, the microcontroller operating the radio module via Serial Peripheral Interface (SPI) can be programmed to process the incoming data packets from the master radio in a very deterministic way. Specifically, the packet arrival interrupt request (IRQ) signal from the radio module causes the MCU to interrupt its current routine and execute a function that decodes the latest timestamp from the data payload of the packet and phase-adjusts the MCU's internal timer to match the master radio's clock. The jitter of the continuously adjusted slave clock is less than 1 μs with the nRF24L01+ 2.4 GHz radio module. The timer, in turn, generates a special synchronization signal connected to one of the inputs of the MCHStreamer device that we use as an audio interface. The MCHStreamer device supports up to 16 channels of synchronous Pulse Density Modulated (PDM) audio recording. An example of how this synchronization signal appears in PCM audio format (converted to by MCHStreamer) is shown in Figure 3.6.

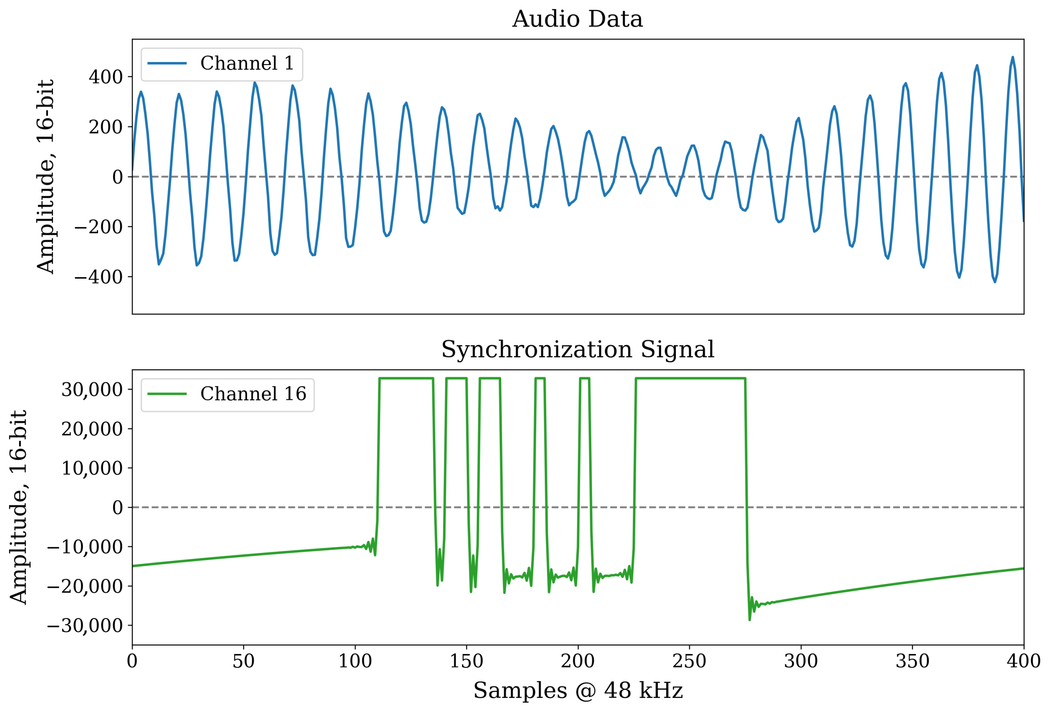


Figure 3.6: Example of a synchronization signal embedded into the last channel of audio data at a 120 Hz rate. It contains a serialized 32-bit timestamp that is shared across multiple sensors with 1 μs accuracy using a 2.4 GHz radio module. High synchronization accuracy is required due to a high audio sampling rate of 48 kHz.

We are using a simple UART-like serial protocol with one start bit, a 32-bit payload, and a more than 200 audio 200-sample-long stop bit to generate the audio synchronization signal. The start bit and payload bits are five audio samples wide for more reliable encoding. Such a signal is easy to decode during audio processing, and a single audio sample synchronization accuracy is achieved because the start bit of the sequence is aligned with the time of arrival of the timestamp from the master radio, and the microcontroller has a deterministic delay when processing this information. An example of synchronized audio data is shown in Figure 3.7.

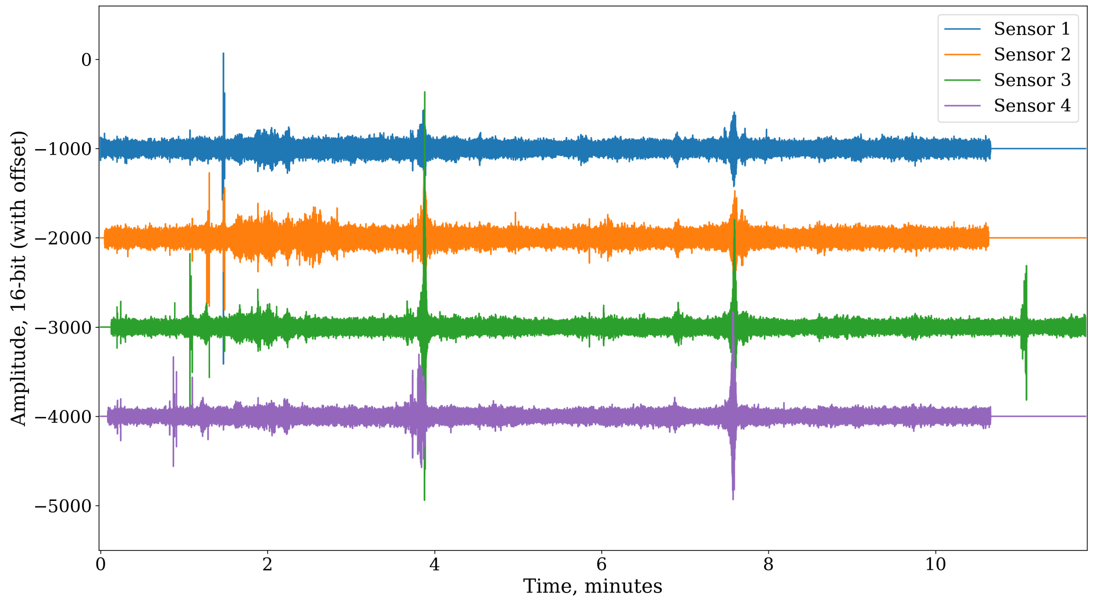


Figure 3.7: A sample of audio data (channel 0) synchronized across multiple sensors. Every 400 audio samples long audio data chunk can be placed in the right place on a global timeline by decoding the serialized timestamps embedded in the dedicated channel at a rate of 120 Hz.

3.3.3.2 Video

Video recording occurs at a much lower sampling rate than audio. For instance, the cameras in REIP sensors are configured to record at 15 fps. That corresponds to a ≈67 ms period between consecutive frames. The radio module receives a new global timestamp every ≈0.83 ms, which is almost two orders of magnitude more frequent. Therefore, it makes sense for video recording to timestamp each frame as it is being received by the driver and calibrates the latency between the moment of assignment of this timestamp and when the frame is actually exposed instead of inventing a way of embedding the timing information directly into the image data during exposure as we did for audio. However, this approach comes with new challenges, such as timestamp jitter and lost frames.

There are three timestamps assigned to each video frame: (1) the GStreamer timestamp, which starts from zero and is defined by the camera driver upon arrival of the frame into the queue from the USB, (2) the Python timestamp representing the current system time, which is added using the time.time() function when the frame is released by GStreamer into the data acquisition and processing pipeline powered by REIP SDK (Figure 3.4), and (3) the Global timestamp added to the frame metadata at the same time as the Python timestamp which is the latest global timestamp communicated to the global time block from the MCU via USB 1.1 interface, introducing extra jitter. Figure 3.8 depicts the jitter progression as it propagates farther down the data acquisition pipeline.

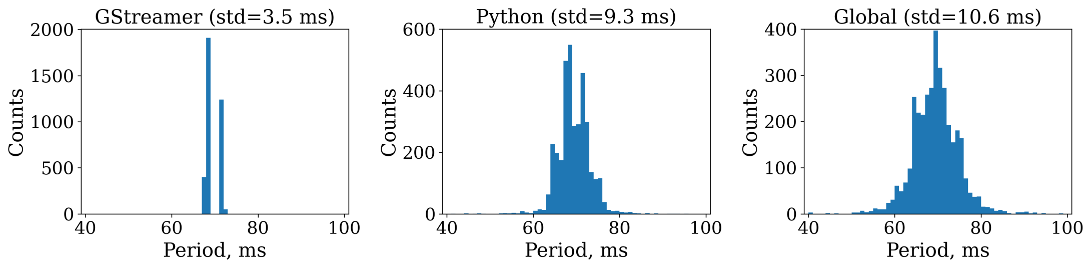


Figure 3.8: Each frame acquired by the cameras is timestamped three times: (I) by the camera driver (GStreamer), (ii) by the REIP framework (Python), and (iii) by the microcontroller receiving global timestamps from the master radio (Global). This figure illustrates a progressive degradation of timestamp quality, in terms of jitter, accumulated throughout the data acquisition pipeline.

We developed a method for reducing the jitter of global timestamps to virtually zero before rendering the synchronized video streams (Figure 3.9). The main source of timestamp jitter is operating system interrupts that happen when the computing platform, for example, needs to process various I/O events or perform memory management. That is why GStreamer timestamps have the least amount of jitter because they are defined when the OS handles USB 2.0 data transfers from the camera. That is also why we are starting with GStreamer timestamps to reliably detect if and when there are any frames lost by looking for gaps larger than the expected period of the camera's frame rate. After correcting for lost frames, we then convert these timestamps into a global timeline through a couple of regressions incorporating the information from other types of timestamps without adding jitter.

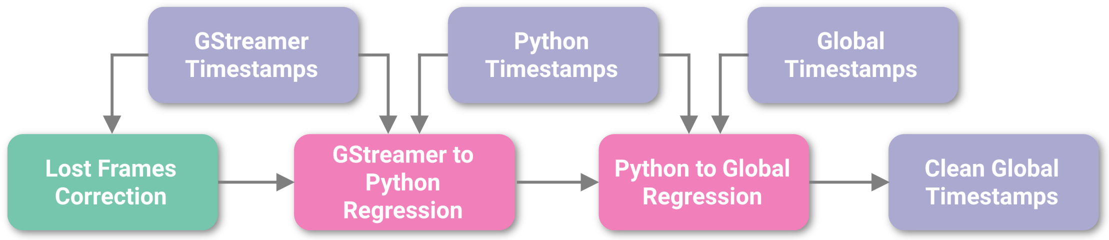


Figure 3.9: Diagram illustrating timestamp processing. We start with the least jittery GStreamer timestamps and identify any lost frames so that we can reconstruct the original timeline and average period for the saved frames. We then convert these reference timestamps into a global timeline through a series of regression steps that incorporate the information from other kinds of timestamps without adding jitter.

In addition to correcting for lost frames and eliminating jitter, our method also fixes any queue overflow issues that often result in the jamming of multiple frames one after another with very similar Python and Global timestamps. This happens when the queue is emptied out quickly after a prolonged operating system interrupt. Another less common issue is when the frames saved to the disk get corrupted due to high data flow or during the copying of the data from the sensors to a server. The solution requires the corresponding timestamps to be deleted from the metadata, and the associated non-decodable frames are considered lost.

To further validate the video synchronization, we render a surveillance-style mosaic video using processed frames from all eight cameras at a given intersection and a global timeline produced by the synchronization of timestamps. Figure 3.10 shows a mosaic of the frames at the moment at the Chase Center intersection. Essential for many analysis applications, at any given moment, the recording of all traffic remains in sync from multiple viewpoints. Frames for which a camera did not successfully record data are temporarily made black in the camera's associated block in the mosaic.

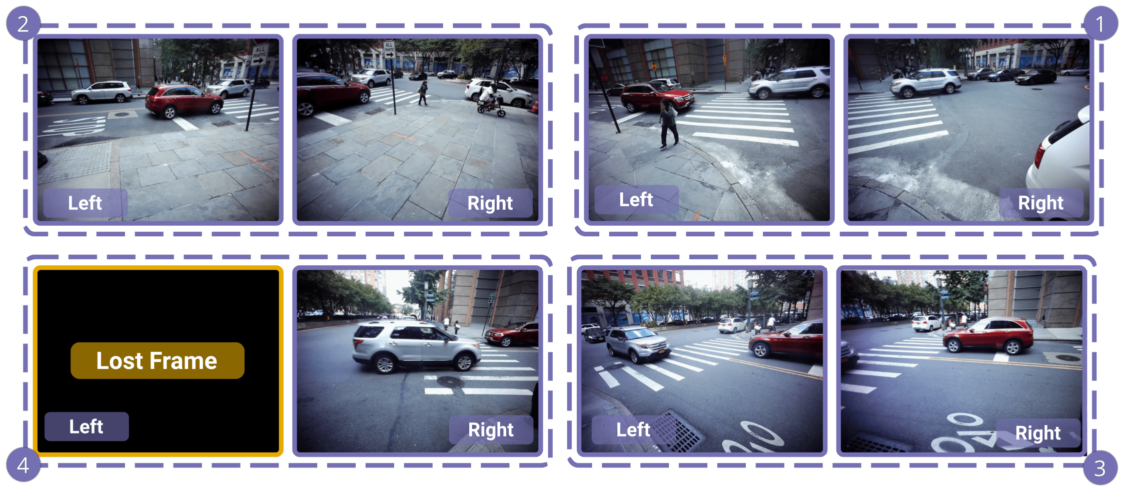


Figure 3.10: Mosaic rendering of the synchronized frames from recording session one at the Chase Center intersection that can be played as a video. Four sensors with two cameras each (numbered in the corners) provide eight different views for comprehensive analysis of the intersection. If a camera did not successfully record during a particular frame, its block is turned black, such as the left camera of sensor 4 in this example.

3.4 Use Cases

In this section, we will demonstrate four use cases highlighting the potential applications of StreetAware. First, we present two examples of how such data can enhance pedestrians' safety in large urban areas by (1) informing pedestrians and incoming traffic of occluded events using multiple sensors and sound-based localization, and (2) associating audio events (such as the presence of loud engines) with their respective visuals. Second, we present easily quantifiable metrics that can be extracted from the data using the StreetAware infrastructure framework: (3) calculating object counts (occupancy) over time, and (4) measuring pedestrian speed during crosswalk traversal.

3.4.1 Audio Source Localization

As the number of sensors deployed in urban environments increases, cities have the potential to become more human-centered by prioritizing pedestrians over cars. Adaptive traffic and pedestrian signal timing is one example of how an intelligent sensing platform can be used to provide a safer environment for pedestrians. By making the signal timing adjustable to the volume of foot traffic as well as the needs of different groups of people, we can allocate longer signal timing to, for example, crowded intersections or pedestrians with special needs such as the elderly, pregnant, or those with vision impairments [78]. Most traffic monitoring systems use one or two fixed cameras for each intersection. However, the complex configuration of intersections in large cities makes it challenging for one or two cameras to count and detect every traffic participant at a busy intersection. They have inherent limitations of fixed field of view and susceptibility to occlusions.

In this first use case, we demonstrate how a synchronized multisensor setting can leverage a data modality, such as audio, to localize sound-emitting traffic participants and reduce the chance an object is completely obstructed by another. The ability of the sensors to "listen" as well as "see" allows the sensor network to remain resilient against occlusions and dead zones. Figure 3.11 shows an example of detecting the position of a bicyclist using sound, regardless of whether or not the bicyclist is in any of the cameras' field of view, thanks to the diffraction property of the sound waves. In order to reconstruct the position of the bicyclist ringing the bell, we first annotate the high amplitude peaks, t_i, in the audio data, synchronized using the common time scale as reconstructed from the dedicated audio channel with the serialized timestamps (see Section 3.3.3.1). With the known sensor positions, p_i, one can find the sound source position, p, at time, t, by minimizing the errors:

$$p, t = \arg\min_{p,t} \sum_{i=1}^{4} \left(||p - p_i|| - c \cdot |t - t_i| \right)^2, \tag{3.1}$$

where $c = 343$ m/s is the speed of sound in air. All four sensors must hear the sound for this to be a well-posed problem. The results are shown in Figure

3.11 and are in good agreement with the video footage from the same sensors. There are examples of when audio-based localization was not possible because of noise pollution by a bus and vice versa when the object was out of the field of view of the cameras but could still be heard which illustrates the benefits of such a complementary multimodal approach. This audio-based sound source localization would not be possible without the synchronization technique presented in this paper.

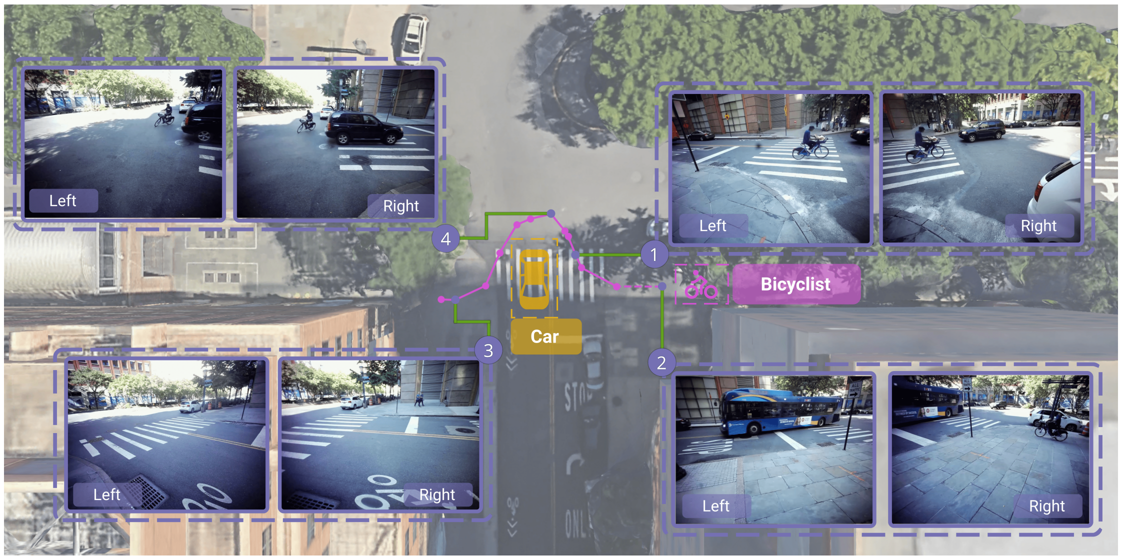


Figure 3.11: Audio-based localization of a bicyclist crossing the street at Chase Center and ringing the bell repeatedly (magenta points). In chronological order: Sensor 2 can see the bicyclist approaching the intersection, but localization of the bell ring is not possible because two sensors were occluded by a noisy bus; Sensor 1 view confirms the position of the bicyclist taking a right turn; Sensor 4 footage reveals the reason for the bicyclist's curved trajectory—the black car did not stop to yield the right of way; Eventually, the bicyclist is no longer in the field of view of Sensor 3, but can still be localized thanks to the diffraction of the bell's sound waves.

3.4.2 Audiovisual Association

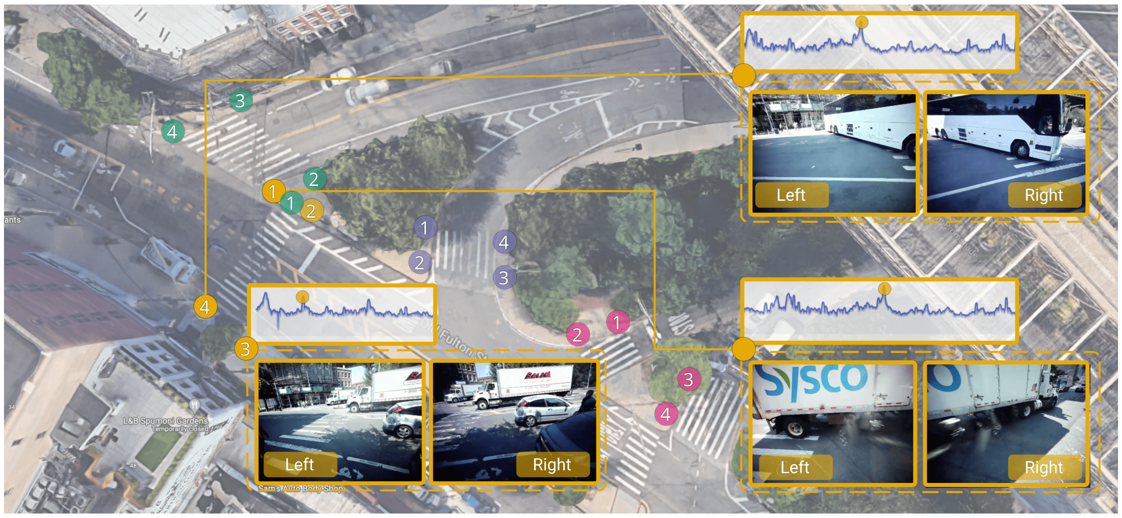


Figure 3.12: Sensor positions during data acquisition at DUMBO, Brooklyn. Colors indicate recording sessions, and numbers denote the sensor. Above each synchronized video frame, we highlight the relative data point in the audio time series (in decibels). It can be shown that events that can be seen in the video, such as the passing of a bus, have a corresponding peak in the audio data.

Automatic audiovisual urban traffic understanding is a growing area of research with many potential applications of value to the industry, academia, and the public sector [80]. Deep learning algorithms can leverage video recordings to detect and count a variety of objects in a given scene and calculate specific metrics, such as the distance from one source to another. Although very useful, these algorithms can be improved through augmentation with non-visual data, such as audio. For example, scene understanding can be improved by determining the proximity of out-of-view objects emitting sounds or by detecting loud noises. In addition, local governments may care about noise levels. In New York, for example, city agencies have created laws to automatically monitor and mitigate noise pollution, such as the noise emitted by loud mufflers installed on cars [47]. Thus, sensor networks that include audio, such as the one outlined in this paper, can provide the audio resources necessary to improve urban scene understanding and to monitor city noise.

With StreetAware, in Figure 3.12, we show how audio can inform the presence of large engines (trucks and buses) at an intersection. Above the video frames, we highlight the corresponding point in time on the acoustic time series (in decibels) extracted from the audio files. With this method, we can easily relate noise peaks to events captured on video. This example shows how StreetAware can advance the state-of-the-art development of audiovisual urban research by providing multiple camera views linked with audio signals to enhance audiovisual recognition algorithms (which are usually trained on single-view video datasets).

3.4.3 Occupancy Tracking & Pedestrian Speed

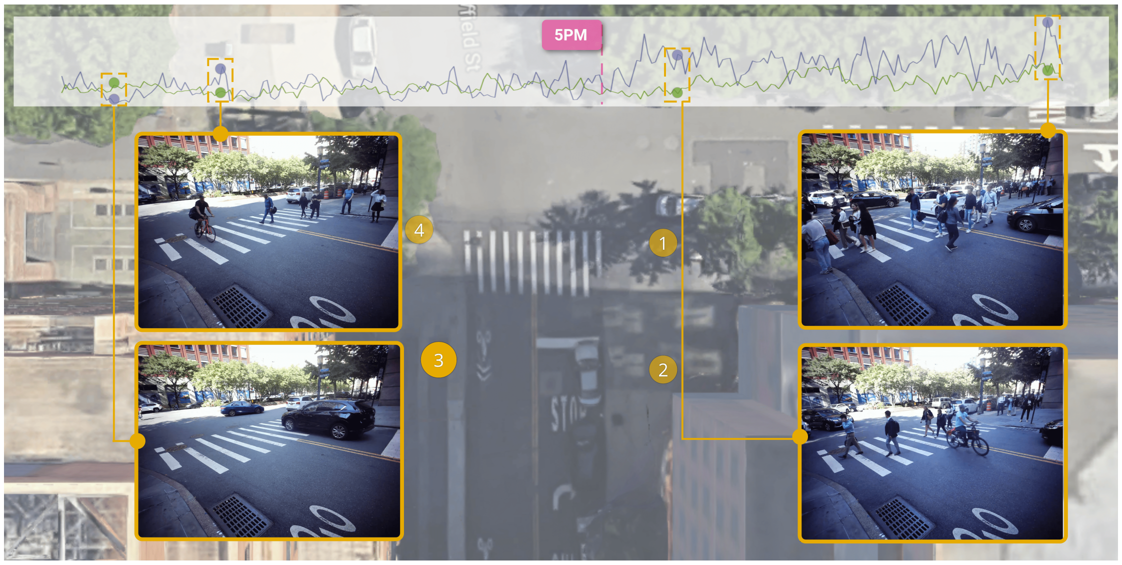


Figure 3.13: Chase Center intersection occupancy by object type during a recording session in the afternoon, with purple lines representing people and green lines representing cars on the top chart. In the figure, the four sensors collecting data during this session are represented by circles. There is a significant (≈3×) increase in the pedestrian count (blue) around 5 p.m. as people leave work. Moreover, it is possible to detect traffic light cycles based on the ratio of the number of pedestrians versus vehicle counts.

Stakeholders interested in monitoring the level of activity and quantity of pedestrians and traffic in an area could make use of StreetAware. Here, we present an example in which we evaluate the occupancy of one of the intersections during a recording session. First, the dataset is evaluated with HRNet, an object detection, human pose estimation, and segmentation algorithm. Adapted from the Faster R-CNN network, HRNet is capable of performing state-of-the-art bottom-up segmentation via high-resolution feature pyramids [235]. The network is trained on the COCO dataset [127]. We detect six classes: person, car, bicycle, truck, motorcycle, and bus (Figure 3.13). Figure 3.14, in turn, shows example visualizations containing detected objects and human pose outlines. For pose estimation, the model is executed for each "person" detection independently with a focus on that particular bounding box. Such an approach results in temporally consistent pose estimation as the person is walking towards or away from the camera despite significant lens vignetting and brightness variation across the image.

Using this detection framework, Figure 3.13 shows the total count of the various urban scene entities throughout an entire recording session at the Chase Center intersection. We intentionally chose this particular recording session because it was conducted around 5 p.m., when people are finishing their workday and traveling home. This activity results in a spike in pedestrian and car traffic. There are roughly three times as many pedestrians counted (most crossing a street) toward the end of the recording than at the beginning. This trend also inversely correlates with car count, presumably because cars yield the right-of-way to pedestrians. We do not observe as much change in the number of cars or other motorized vehicles because this intersection is typically more consistently busy throughout the day, and there is limited space along the street curbs to park cars compared to pedestrians on sidewalks. Parked cars present a certain level of static background count for the car object class.

As outlined in Section 3.2.2, measuring and predicting pedestrian behavior, such as their speed and trajectory, are of interest to the computer vision and urban design research communities. Figure 3.14 presents a simple example of capturing the same pedestrian across two different sensors, highlighting the utility of multiple camera views. The speed of the pedestrian in Figure 3.14 is manually calculated at 1.1 m/s, derived from traveling $\approx$11 m (as measured from Google Maps) in 10 s (12,000 global timestamps difference at 1200 Hz up-

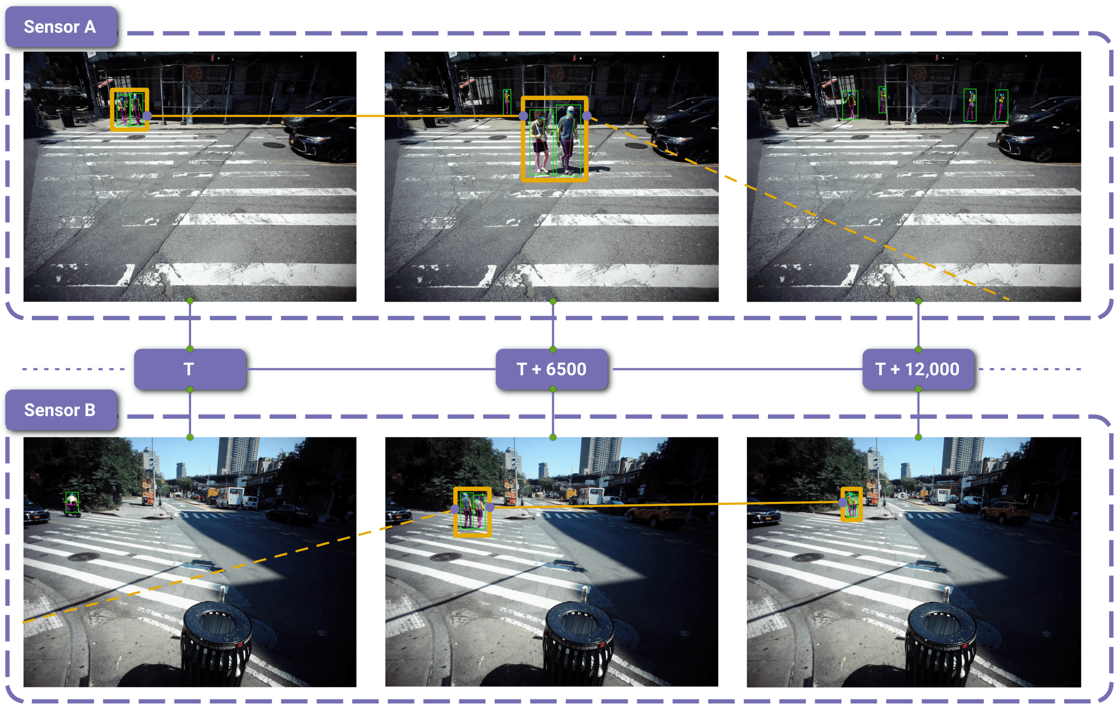


Figure 3.14: Two camera views from the session 2 recording at DUMBO. Camera A points northwest, and Camera B points southwest. At the time, T pedestrians (surrounded by an orange box) are visible in camera A but not in camera B. At the time T + 6500, as the pedestrians cross the street, they are observable by both cameras. By T + 12,000, the pedestrians are no longer observable in camera A but are still visible in Camera B as they continue to walk down the sidewalk. Time T represents the global timestamp at the moment the pedestrians begin crossing the street. By extension, T + 12,000 is the time 10 s later because the global timestamps are updated at a rate of 1200 Hz. This figure also highlights the advantage of high-resolution video. With objects at a farther distance from the camera, it becomes more challenging to detect them and estimate their poses. Higher resolutions help mitigate the information loss associated with more distant objects occupying a smaller portion of an image.

date rate). Therefore, one could use a deep learning algorithm and the data's internal timing to accurately and automatically measure pedestrian speed.

3.5 Final Considerations

In this study, we collected unique data about traffic and pedestrians from three urban intersections using customized high-resolution audio-video sensors. The novel data includes multiple modalities (audio, video, and LiDAR) with highly accurate temporal information and synchronization. Since the data were recorded in New York City, many demographics are captured. This is particularly important since some of these groups, such as wheelchair users and people with varying levels and types of disabilities, are absent from large-scale datasets in computer vision and robotics, creating a steep barrier to developing accessibility-aware autonomous systems [264]. Identifying pedestrians with disabilities, the qualities of their behavior, and ease at traversing the sensed urban environment is an area of possible exploration with datasets such as this one.

With high-resolution video data, such as in StreetAware dataset, it is important to protect people's privacy. For that, we leveraged human pose detections to identify where pedestrians are and applied Gaussian blur over the elliptical areas covering their faces. Because automatic methods are not perfect and complete pose detection is particularly susceptible to misdetection in highly crowded areas due to occlusions, we also employ a second model that does direct face detection [251]. Combined with aggressive detection thresholds that result in a high likelihood of producing false positives, we were able to achieve robust video anonymization across the entire dataset.

In Section 3.4, we demonstrated four uses of the data that are not possible with many other datasets. Section 3.4.1 provided an example of how the multiple perspectives and audio data can be leveraged for the localization of the sound-emitting objects to help overcome visual occlusion by other objects such as large vehicles. Section 3.4.2 showed that there exist qualitative associations between objects captured by the sensors' cameras and sounds captured by the sensors' microphones. More quantitatively, in Section 3.4.3, we showed that a computer vision model can track the amount and type of objects in our data, confirmed by checking the video and counting numbers at specified times. With closer inspection of the occupancy data, one can notice a regular pattern in bus occupancy. Indeed, a bus route does pass through the intersection. This further highlights the importance of our synchronization technique with diligent temporal accounting to correct for any lost frames that might accumulate into a significant time gap in the video footage. Otherwise, attempts at temporal analysis, such as, for example, reconstruction of the bus schedule, would suffer from a systematic error.

Finally, in Section 3.4.3, we presented a simple example of how a coordinated arrangement of multiple synchronized cameras can provide a foundation for pedestrian tracking applications, i.e., unique detection of pedestrians consistent across frames and views. Some currently available software, such as NVIDIA's DeepStream SDK [179], contains built-in C/C++ and Python pipelines for pedestrian tracking. Such tracking technologies could be combined with geo-referenced locations for pedestrians and vehicles to create a map. This digital twin of an intersection, complete with object locations, can be used for high-level analysis such as determining pedestrian and vehicle counts, travel distances, speeds, and trajectories as they navigate their way through the sensed space.

3.5.1 Limitations

Overall, the REIP sensors have demonstrated great versatility in data acquisition pipelines and operating conditions. They even withstood, without damage, a sudden rain incident during one of the recording sessions at Commodore Barry Park. The majority of the sensor's hardware is enclosed within an aluminum weatherproof housing with heat sinks, however, we still experienced occasional periods of lost frames, even during operation in shadows, due to the random operating system interrupts and throttling of the CPUs. Therefore, any long-term deployments would need to account for these issues in a comprehensive way.

The data presented in this study are limited in a few ways. First, the geographic coverage is narrow. Though the activity at each site is somewhat varied, ultimately, data were only collected at three intersections in a single borough in a single highly-developed city in the United States. Second, compared to some other available datasets, StreetAware lacks diverse environmental conditions such as nighttime, precipitation, and fog. However, we did preserve some of the more challenging recording sessions where select sensors experienced an increased amount of occlusion from vegetation during windy conditions. Moreover, third, the data are quite raw—the audio and video recordings are not labeled (e.g., objects, actions, sound sources, etc.) and the LiDAR files provided are unprocessed. In its current form, a user would not be able to query the data for information or have an idea of what is happening over time in a scene without manually inspecting the data or performing further processing and analysis.

3.5.2 Conclusion

In this chapter, we presented the StreetAware dataset, which contains synchronized multi-perspective street-level audio and video in a single dataset. We also presented a new method to synchronize high-sample rate audio streams and demonstrated unique use cases for the data; in the process, we describe the limitations and real-world implementation of REIP sensors.

Moving forward, further applications can be developed to make use of a digital map, such as calculating the distance between vehicles and pedestrians and other vehicles and, thus, the detection of near-collision situations. Aspects unique to StreetAware, such as audio, LiDAR, and multiple in-sync views could be used to augment the performance of such applications (e.g., incorporating the sound of car horns into near-accident detection). Other future areas of investigation include determining the optimal number of cameras to capture the same information captured in this dataset, and the viability of processing the data in real-time on-site (edge computing). Building off the pedestrian detection and speed measurement established here, looking ahead, we intend to evaluate pedestrian and vehicle movement per traffic light cycle. We will leverage the multi-view and synchronization features of the dataset to reconstruct the timing of traffic lights as seen from different camera locations. This will enable us to measure pedestrian and motorist adherence to traffic laws. Other researchers exploring urban street sensing applications that benefit from high-resolution, multimodal, and precisely synchronized data should find this dataset especially useful.

Chapter 4

Crossroads: A Pedestrian-Centric Visual Analysis of Crossing Dynamics in Urban Environments

4.1 Introduction

The analysis of urban intersections and pedestrian crossings has been an integral part of transportation studies for decades. Despite their seemingly simple design, intersections embody significant complexities, specifically from the traffic engineering and planning perspectives. This complexity extends across various dimensions, including design, control mechanisms, and considerations for safety, right-of-way, and traffic flow [112, 163, 218]. Each of these aspects intertwines to create multifaceted challenges that require a deep understanding and sophisticated solutions to effectively manage and optimize intersection performance. As a shared space between motorized and non-motorized users with varying operation speeds, mobility levels, and reaction times [115], intersections are known to pose higher risks of accidents, specifically for more vulnerable users such as pedestrians and cyclists.

Urban crossing regions and intersections play a critical role in urban circulation and dynamics; however, many studies have predominantly focused on optimizing these spaces to reduce car traffic congestion and improve vehicular flow, often at the expense of pedestrian needs and safety. Engineers and planners frequently use advanced quantitative methods to analyze intersections, prioritizing signal systems and timing adjustments that favor vehicle throughput over pedestrian considerations.From a planning perspective, there is increasing recognition of the need for a more balanced approach that equally prioritizes pedestrians' and cyclists' safety and convenience [112, 163, 227].

Shifting towards a pedestrian-centric approach for the analysis and planning of intersections poses several challenges, particularly in data acquisition and analysis. Traditionally, intersection analyses rely on incident-level crash data, which often fail to capture the complete context of accidents and do not provide sufficient information for thorough safety assessments and effective prevention strategies [112]. Advanced approaches that utilize video footage typically depend on top-down camera views [123], which obscure crucial micro-decisions and behavioral patterns that are essential for comprehensive safety analyses [82, 102, 137, 171].

The process of analyzing these videos is not only time-consuming but also prone to inaccuracies, which can undermine the identification of critical patterns and anomalies. In fact, manually inspecting various agents — pedestrians, cyclists, vehicles, *etc.* — within these complex environments, and analyzing their behavior is impractical. To tackle these challenges, we introduce Crossroads: a visual analytics system that leverages a novel multimodal dataset [185] composed of audio and video collected from various urban crossings in New York City. While many existing approaches focus on high-level aggregated metrics or single-modal data, Crossroads addresses the complexity of analyzing micro-level interactions and agent behaviors in a spatiotemporally dense, multi-modal context. By seamlessly integrating video, audio, traffic signal states, and 3D reconstructions, our system confronts the unique visualization hurdles posed by noisy, real-world intersection data. Aligning these heterogeneous data streams to reveal both global trends and fine-grained incident-level patterns requires a specialized design to handle occlusions, sparse views, and the need for precise user-driven corrections.

At the core of Crossroads are: (1) an automatic data enrichment strategy used to add semantics to urban crossing footage, such as physical measurements of the actors present at the crossing region; (2) a human-in-the-loop trajectory augmentation and refinement approach used to better understand agents' movements; and (3) a set of interactive visualizations that enable the analysis of urban crossings.

Automatic Data Enrichment Crossroads's enrichment strategy involves the identification of crossing agents, the computation of agent trajectories, and the classification of sound types (sirens, horns, *etc.*) synchronized with traffic light states. By adding this layer of semantics, users can efficiently and expressively build query events within long video recordings, making it possible to filter hours of footage for meaningful events and streamline subsequent analysis. As part of this process, we employ an enrichment pipeline based on modern computer vision techniques to automatically locate, track, and reconstruct objects of interest (e.g., pedestrians, cyclists, cars, and trucks).

Human-in-the-loop Reconstruction The human-in-the-loop data refinement approach used in the system improves the 3D representation of the agents interacting in the scene, adding a new layer of information to the dynamic patterns visible in the videos. Reconstructing urban crossings with real-world footage is particularly challenging due to noisy, sparsely overlapping frames that often hinder feature matching and structure-from-motion techniques [205]. By enabling users to supervise the reconstruction process, Crossroads improves the reliability of the metrics extraction process, ultimately yielding more accurate trajectories and visual representations of agent movement. This interactive feedback loop empowers domain experts to directly address uncertainties in automatically generated data.

Interactive Visual Analysis Finally, the system incorporates a set of interactive visualizations that support both macro-level overviews and micro-level interrogations of urban crossing behavior. Planners and researchers can compare events, observe global agent behaviors, and examine individual trajectories while benefiting from the real-time feedback loop that ensures high-precision analytics. This approach not only supports established analytical needs — such as aggregated speed or turning counts — but also facilitates nuanced, instance-level exploration of trajectory anomalies, near-collisions, or traffic violations. By unifying multiple data modalities in a single visual interface, Crossroads bridges the gap between conventional top-down analysis and fine-grained, user-driven corrections. In summary, our contributions include:

- An automatic data enrichment pipeline to extract semantic information from a large collection of video footage. The output of this pipeline is a digital twin representation of the intersection, enabling spatiotemporal queries that can capture events and retrieve summary statistics of the various crossing regions.
- Crossroads, a visual analytics tool that comprises a set of interactive and interlinked visualization metaphors to allow domain experts to pose queries and inspect results to identify events and global behavior patterns in urban crossings. Moreover, our tool also supports user supervision to extract and refine accurate 3D information from the crossing agents.
- A set of usage scenarios developed together with a domain expert, leveraging both summarization capabilities of Crossroads and detailed inspection of specific high-risk scenarios. These case studies highlight how our tool can help detect infrastructure problems that can harm the safety of such intersections.

4.2 Related Work

In this section, we first review papers analyzing pedestrian behavior in dynamic environments. Then, we present previous works on urban visual analytics systems targeting multimedia (video and audio) data.

4.2.1 Urban Crossroads

According to the World Health Organization, about 70% of all pedestrian deaths in European Union countries and 76% in the United States occur in

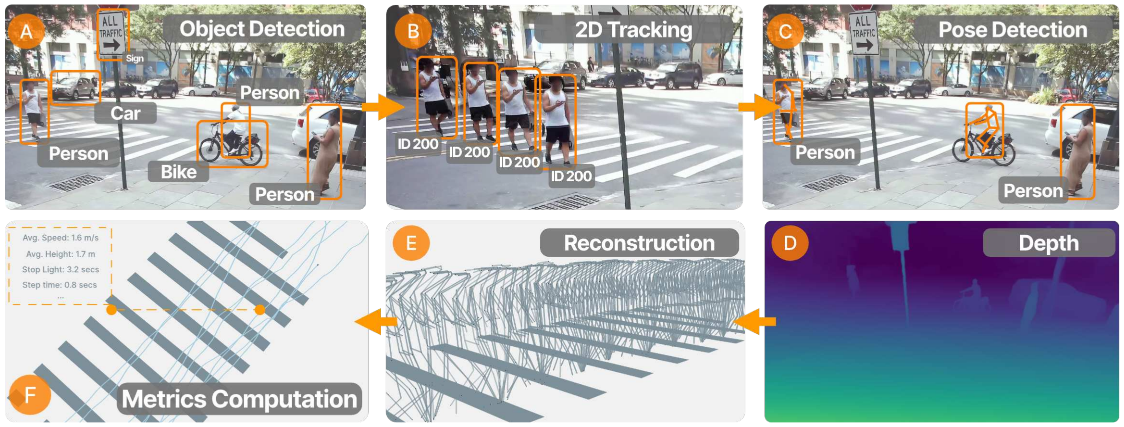


Figure 4.1: Crossroads's Data Processing Pipeline, broken down into six steps: Operations in 2D include Object Detection (A), 2D Tracking (B), and Pose Detection (C). Once these operations are done, the pipeline proceeds to operations in 3D space. First, depth computation is computed for each video frame (D). With that information, we can then reconstruct different actors (such as pedestrians and cyclists) in the 3D urban space (E). Finally, once the urban space and actors are reconstructed and tracked in 3D, we automatically compute and display different metrics, such as average speed and time taken between steps (F).

urban areas [181]. Safety concerns often arise in road areas where different types of transportation (e.g., passenger cars, public buses, bikes) compete for space, sometimes in the same lane [136]. The rise in popularity of electric scooters and electric bicycles this past decade has been a cause of alarm in pedestrian safety: researchers are studying pedestrian injuries related to these electric modes of transportation, which achieve higher speeds than, say, a normal bicycle, and with many riding them on the sidewalk [98, 212]. In the United States, for instance, pedestrian deaths increased by 41% between 2008 and 2018 [181]. In 2021, 22% of pedestrian deaths in the United States occurred at intersections [104]. Intersections are inherently high-risk locations where various transit modes converge. Additionally, with the ubiquity of mobile phones, their usage on the streets is linked with decreased attention and increased injuries and fatalities [168, 169]. Wells et al. describe a process where coders physically stood at busy intersections on two college campuses to document and investigate the impact of distractions (e.g., texting, talking on the phone, wearing headphones) on street-crossing safety [244].

An important task related to analyzing intersections involves implementing effective modal separation strategies. Often, these strategies are heavily skewed toward facilitating vehicular traffic, such as aiming to maximize flow and minimize delays for vehicles [136], overlooking the risk to more vulnerable users such as pedestrians and cyclists. Recent studies have attempted to address this shortcoming by incorporating pedestrian movement and vehicle interaction patterns into analyses [44, 66, 176]. Recent research has also focused on developing a better understanding of the nature and challenges surrounding the issue of pedestrian safety [83, 259]. Many of these studies focus on pedestrian behavior in relation to intersection design [207]. There is evidence that pedestrian expected waiting time influences the number of attempts needed to cross a street [22] successfully and the collision risk [9]. In a study of 1864 intersections in Montreal, Canada, it was found that curb extensions, raised medians, exclusive left turn lanes, and pedestrian priority phases of the traffic light reduced pedestrian injuries [218].

To perform intersection safety analysis, first, data (e.g., audio & video) collection of intersection participants is required, followed by organizing the data into usable datasets [208]. The data can then be used to model pedestrian behavior and safety. Models, such as linear regression and neural networks, are chosen based on the modeling task and the size and variation of the data. For example, Quintero et al. [191, 192] used a 3D pose of pedestrians to learn pedestrian dynamics in a latent space for predicting actions and directions. From there, such modeled patterns can provide a higher-level understanding of behavior and safety, such as examination of the trajectory (e.g., calculating the crossing speed of pedestrians [228]) to more complicated analyses (e.g., predicting a driver's turning intention or the development of a safety measure such as time-to-collision [13]). Given the importance of easily viewing and navigating intersection data and the output of associated models, Crossroads aims to provide a consolidated way to efficiently search for and visualize various forms of analyses within an intersection dataset — all without the need for manual data collection. Crossroads will also help validate findings and test hypotheses similar to those in the aforementioned studies.

4.2.2 Urban Analytics for Multimedia Data

Our work is situated at the intersection of urban visual analytics and multimedia analytics. In urban visual analytics, several previous contributions have been made to enable the analysis of data generated within the boundaries of cities and to tackle urban-specific problems [59, 71, 73, 272]. Most of these works, however, make use of tabular data, such as taxi trips [75] or social media activities [152]. With the advancement of sensing capabilities, recent works have also leveraged multimedia data for urban analysis, leveraging datasets composed of images [154], laser scans [94], audio [202], and surveillance videos [20, 121]. However, most works in the surveillance category propose to (1) use visual analytics to improve computer vision tasks for video data, such as improving detection of traffic light [90], movable objects [99], or comparison of models [106]; or (2) use visual analytics on features extracted from raw video, without allowing the user to inspect the videos themselves, such as Bi et al. [23], that used video data to simulate intersection traffic, and Sun et al. [220] that used features for traffic surveillance. Few visual analytics studies utilize video data to address traffic congestion while still allowing users to inspect and correct errors in the data visually. For example, Lee et al. [121] incorporated CCTV video data for real-time surveillance and prediction of traffic congestion. Piringer et al. [187] proposed AlVis to explore live and historic CCTV data in tunnels. Namitha et al. [166] proposed an interactive tool to realize queries to generate surveillance video synopsis. In contrast, Valdrighi et al. [233] proposed MoReVis to incorporate visual encoding to enhance spatial context in 2D videos and analyze trajectories and interactions between agents in 2D space.

Our work stands out from previous works in two key ways. First, while previous works have primarily focused on car-centric tasks and problems, our work is the first to use street-level video data within a visual analytics system. Our goal is to better understand the dynamics between pedestrians and various urban elements (e.g., cars, bikes, traffic lights). This approach represents a shift towards pedestrian-focused urban analysis, a significant departure from the more common vehicular perspectives in the field. Second, earlier efforts have relied on high-quality data obtained from extensive sensor networks and have filtered out noisy, irrelevant data. Instead, our study uses data from cost-effective sensors, integrating both video and audio data. We provide functionalities in Crossroads to tackle their inherent problems, allowing users to not only filter out noise but also improve data quality through a visual feedback loop. This, therefore, enhances the applicability of our findings in real-world settings.

4.3 Challenges and Requirements

In collaboration with urban planners and transportation researchers, we have identified a series of challenges in designing a visual analytics system tailored to the study of intersections. These challenges emerged through a series of meetings, during which experts from both domains shared their insights into analyzing intersections. First, since intersections are complex environments, fully capturing their dynamics requires integrating data from different modalities, such as audio and video recordings. Second, given the size and complexity of these data, it would be unfeasible to manually inspect them directly to outline the characteristics of an intersection or retrieve events such as bike-pedestrian collisions to perform an incident analysis. To enable effective intersection analysis, the system must allow users to compose expressive queries across multimodal datasets. The more expressive these queries are, the more thoroughly users can explore the crossing data, uncovering insights to inform new interventions and shape urban policies.

Additionally, queries and analyses may depend on information that is not directly measured by cameras or sensors but can be derived from the sensed data. For instance, pedestrian, cyclist, and vehicle trajectories and their velocities in a 3D representation of the intersection can be reconstructed from video footage, enabling the identification of risk situations such as near misses. Likewise, recognizing audio patterns, such as horns and sirens, can help identify periods with valuable contextual information. However, automatically extracting 3D trajectories from data captured in complex and uncontrolled environments, such as intersections, is challenging. Variations in light conditions, occlusion, noise, and other recurring issues can cause off-the-shelf algorithms to present high failure rates, which can significantly degrade the quality of the extracted information and render it unusable for analysis (see Section 4.5.2 for details). Although recent advances in 3D reconstruction from sparse views, such as Dust3r [237], are promising, these models are mostly trained on and targeted towards static scenes of rigid objects. In Crossroads, on the other hand, we are interested in answering questions about the motion and interaction of different agents in urban environments. For this reason, implementing a human-in-the-loop strategy is important to enhance the computer vision and machine learning models used to extract information from multimodal data. This approach also serves as a data consistency validation, ensuring that high data quality standards are maintained. Finally, such a system must offer visualizations and interactions that allow users to create visual summaries of an intersection while also allowing them to flag and annotate data to identify significant events or inconsistencies that need to be addressed.

Based on these challenges, we established a set of requirements to guide the development of Crossroads.

[R1] **Multimodal data queries.** Support the ability to compose and execute queries across video and audio data, enabling system users to correlate information across modalities and understand patterns among the actors' interaction in the region of interest.

[R2] **Automatic extraction of semantic information.** Automatically extract interpretable information from multimodal data to support longitudinal analysis. This includes the detection and tracking of the movements of pedestrians, cyclists, *etc.*, as well as the identification of audio events, such as horns and sirens.

[R3] **Trajectories reconstruction and validation.** Reconstructing the 3D trajectories of crossing agents is vital to analyze their behavior better. Since automatic state-of-the-art approaches still produce noisy results in real-world situations, the system should adopt a human-in-the-loop approach to validate and improve the 3D reconstruction of crossing agent trajectories.

[R4] **Annotation of data elements.** Provide users with the ability to an-

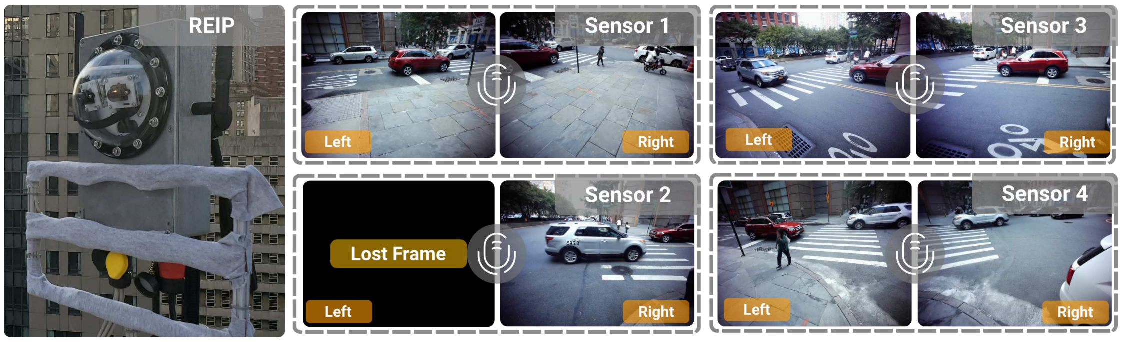


Figure 4.2: StreetAware Data. On the left, we include a picture of the sensor, containing two high-resolution cameras and a 12-microphone array, protected under a layer of gray cloth. The eight images on the right provide an overview of the sensor arrangement at one of the intersections: there are four sensors, each with cameras labeled as left and right. We chose this frame to exemplify the recurring lost frames, which we discuss more in-depth in Section 4.5.2.

notate data during the analysis, allowing them to tag significant events, such as near-miss incidents, 3D trajectories that need refinement, and trajectories for comparison.

[**R5**] **Visual representation of crossing states.** Provide a 3D visual summary of an intersection, capturing global agents' behaviors and spatial patterns and the trajectories and interactions of agents in particular situations.

4.4 The StreetAware Dataset

Our work makes use of a novel dataset called StreetAware [185]. The creation of StreetAware was driven by the recognition that intersections are data-rich nodes within urban environments. These locations capture a wide range of complex interactions between pedestrians, cyclists, vehicles, and infrastructure (see Figure 4.2). The StreetAware dataset consists of approximately 8 hours of audio and video recordings at different intersections in New York City. These intersections were selected in collaboration with transportation researchers to capture a diverse range of scenarios, covering diverse demographics, urban layouts, and built environment profiles. The recording sessions used the sensors specifically designed to accurately synchronize multiple video and audio inputs.

Each sensor is equipped with a 12-channel microphone array that collects audio at 48 kHz, along with two high-resolution video cameras (2592×1944 pixels) capable of recording at 15 frames per second. Together, the cameras in each sensor provide an approximate horizontal field of view of 160 degrees. The cameras are positioned approximately 2 meters above the ground, offering an optimal view of pedestrian activity at the intersections. Each recording session captured approximately 45 minutes of data, with four sensors deployed at an intersection, producing eight synchronized videos from distinct viewpoints. Figure 4.2 provides an overview of an arrangement at one of the intersections. In addition to synchronized videos, the dataset also includes synchronized audio data. The audio data from each microphone was synchronized, enabling accurate classification of audio events and precise extraction of the locations of audio sources. Additionally, the dataset includes the intrinsic matrix parameters for each camera, and all videos have been pre-processed to remove distortion.

In total, the dataset provides 527 GB of synchronized and anonymized audio and video data. The StreetAware dataset has the potential to assist domain experts in a wide range of different analytical tasks, helping them better understand the dynamics of intersections and support more accurate, data-driven decisions. However, the complex nature of the data presents challenges for exploration, as previously highlighted. Querying and summarizing events across multiple data modalities requires visualization techniques to ensure that users with limited technical expertise can interactively extract meaningful insights from the raw data streams.

4.5 The Crossroads System

To address the challenges outlined in the previous sections, we introduce Crossroads. In this section, we first provide an overview of the automatic data enrichment process employed by Crossroads (Section 4.5.1). Then, we present Crossroads's interface and interactions to support querying and comparison capabilities (Section 4.5). Finally, we present a discussion of opportunities for human-in-the-loop interventions to fine-tune and expand 3D information (Section 4.5.2).

4.5.1 Data Enrichment

Processing Crossroads's data to extract physical measurements involves several computational steps, as outlined in Figure 4.1. In the 2D image operations, our goal is to extract a set of consistent trajectories within the image space (i.e., 2D trajectories), representing the movement of the actors interacting on the crossing region, such as cars, pedestrians, and cyclists. The outputs from the 2D space are then passed to the next set of steps, which operate in the 3D urban space, with the goal of extracting trajectories within the urban environment (i.e., 3D trajectories) and then extracting measurements of interest. Finally, a set of operations augments the data with information regarding environmental noise and traffic lights. The produced data is stored and indexed in a relational database to support analytical queries.

2D enrichment To achieve requirement **R2**, our pipeline starts by processing raw video data (2D space) from all of the StreetAware sessions. In this step, we perform three operations: object detection, pose detection, and 2D tracking. First, the pipeline starts by processing the videos to detect the presence of objects of interest in every frame, applying the widely used object detector YOLOv12 [109]. Concurrently, we also track the detected objects through the frames of the video, assigning a tracking ID to each one of the bounding boxes using the state-of-the-art multi-object tracker BotSort [4]. The output of this step is a set of bounding boxes per frame together with their track IDs and class names, allowing us to extract trajectories of the identified objects. Following this step, we extract human poses (HRNet [45]) and car keypoints (OpenPifPaf [119]) from every bounding box classified as such. These will be needed to compute the 3D trajectories and associated metrics described in the following subsection. These steps are represented in Figure 4.1-A, B and C.

3D estimation As outlined in requirements **R3** and **R5**, Crossroads should enable users to query and visualize dynamic aspects of the crossing regions that are only accurately computed in a three-dimensional space, such as speed, acceleration, height, and direction. Therefore, for each one of the trajectories described in the previous step, classified as either pedestrian or car, we recover the three-dimensional information of these actors using the keypoints generated by the pose detection algorithms described above. We accomplish this by estimating depth maps [253, 254] from the frames of the videos and transforming each one of the pose keypoints into a 3D point as represented in Figure 4.1-D and E. Once this process is done, however, the reconstructed information from each video will live in its own coordinate system. So, we transform the points to a shared coordinate system based on the extrinsic camera parameters computed based on the annotations made with the registration helper (more details on this process are provided in Section 4.5). As represented in Figure 4.1-F the trajectories of each video are projected into this shared space representing the intersection region. From these trajectories, we extract metrics of interest that will enable users to search and visualize events happening in the recording sessions. We utilize the 3D pose representation, to calculate measurements such as height, average speed, acceleration, and gait information.

Although these 3D representations of the trajectories can provide an initial estimation of physical measurements, allowing users to query for events, they suffer from the noisy output of the depth maps and potential occlusions, as shown in Figure 4.3. Moreover, this approach generates redundant information since trajectories representing the same agents are reconstructed once for each video, producing redundant data. Due to this fact, Crossroads provides a mechanism (see requirement **R3**) where users can manually match trajectories across the camera feeds to generate more accurate representations of the trajectories, which will be discussed in Section 4.5.2.

Environmental Information Lastly, we also link environmental information with the reconstructed data (see requirements **R1)-R2**. We first use a machine listening model (YAMNet [85]) to detect informative sounds at urban intersections, like car honks and emergency vehicle sirens. To process the audio, we first normalized the sound levels. Then, we split the audio into 2-second segments. For each segment, YAMNet gives us a list of scores that show how likely each audio event is present in that segment. Most recordings in the StreetAware dataset were captured at intersections with traffic lights.

At these intersections, we marked pedestrian signals by drawing a box around them and used the average red brightness to determine if the signal indicated *cross*, *stop*, or *flicker*. This information helps us identify unusual situations at intersections, such as jaywalkers and pedestrian acceleration when the red light is flickering.

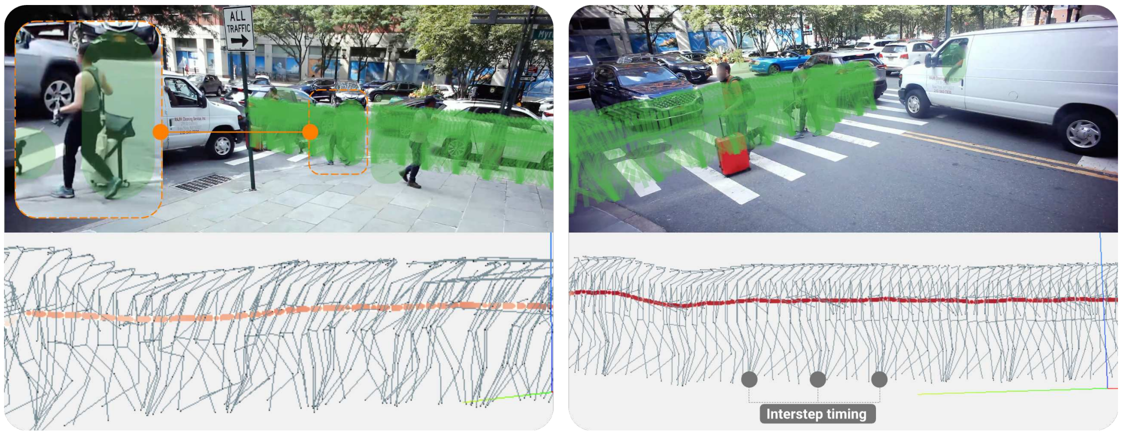


Figure 4.3: Crossroads can recover trajectory information in occluded scenarios due to its ability to handle multi-view information.

4.5.2 Human-in-the-loop Interaction for Data Enrichment

Crossroads deals with real-world data captured in crowded environments, which can introduce various challenges, such as occlusion (e.g., a person may be obscured by a passing car) and environmental factors (e.g., changing lighting). These difficulties can introduce undesired noise in the data and incomplete representations of the dynamic actors interacting in the scene. However, with simple interactions, users can provide valuable information to the system to improve the data and augment the automatic representation generated by our processing pipeline.

This section presents key opportunities to enrich the processed data with user feedback. We first discuss our approach to facilitate camera registration. Then, we present tracking and reconstruction augmentation strategies that

Crossroads implements through intuitive interactions through our interface, satisfying **R3**.

Camera registration One of the main challenges discussed in this work is recovering dynamic 3D information from urban intersections using video recordings from sparse views. This process needs suitable approaches to recover the position of the different cameras in the scene, also known as camera registration. These methods rely on automatic keypoint detection and matching algorithms, which are typically designed to process video data from a moving camera with highly overlapping fields of view. However, the StreetAware dataset is captured from a small set of cameras at fixed locations, each with different fields of view. Classic feature detection and matching algorithms, such as SIFT [133] and FLANN [159, 160, 161] have a high likelihood of false detections and matches under scenarios similar to ours, which negatively impact the calibration accuracy. Throughout the development of this tool, we experimented with different automatic registration approaches to recover camera positions in each recording session. However most of them fail to adequately produce good camera position estimations. We solve this issue by observing that manually annotating matching points on the pedestrian crossing lines should be a straightforward task for users, requiring a marginal effort to produce accurate sets of matching points across different views. With Perspective-n-Point pose computation, a camera can be successfully registered with as few as four non-collinear points. This observation motivated the need for the registration helper view described in Section 2.5, which supports users in annotating matching points across different views. In addition to acquiring a more accurate camera registration, the manual annotation of points in Crossroads supports real distances between the annotated points of the crosswalk zebra. With these measurements, our system recuperates the relative position and orientation of all cameras and places them in a ”world” coordinate system with a meaningful scale. Therefore, distance traveled in a trajectory can be, for instance, expressed in meters or inches, which in turn enables users of Crossroads to better understand and analyze the environment of these urban intersections.

Reconstruction augmentation Although our data processing pipeline relies on state-of-the-art computer vision algorithms to detect, track, and

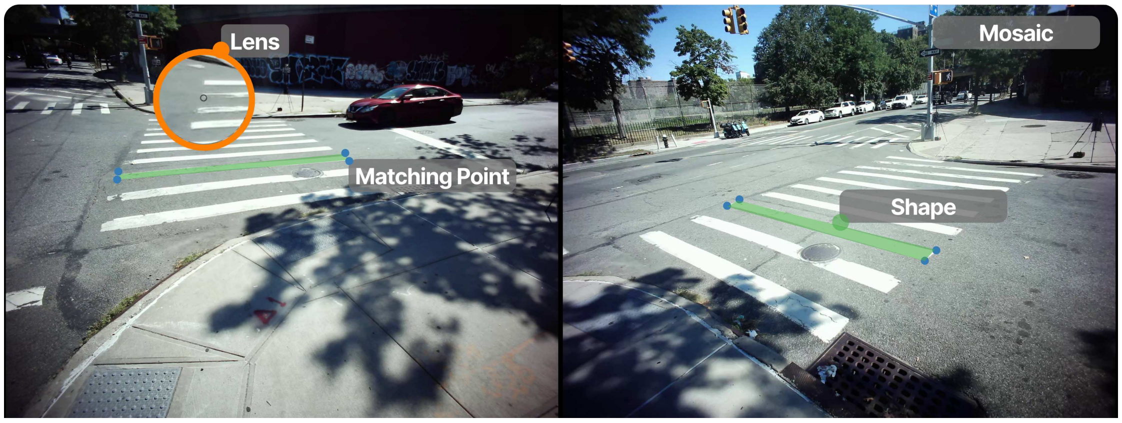


Figure 4.4: Mosaic of frames employed to define matching points from different cameras used in the registration process. To increase the user precision, the system implements a lens-inspired tool.

reconstruct the dynamic information of the crossing regions [221, 236, 256], as any machine learning-based algorithm, these are susceptible to errors. Below, we present two examples of how simple interactions implemented in Crossroads can help users augment the data during interaction time and evolve the dataset over time.

Occlusions frequently happen at busy intersections, mainly in cases of heavy traffic or the presence of large vehicles. These events might hinder the accurate calculation of trajectories, hiding information regarding the pedestrian behavior in these scenarios. Using Figure 4.3 as an example, we can observe the case where a pedestrian carrying a suitcase and a dog tries to cross a busy street. The occlusion generated by a large vehicle prevents us from visualizing the pedestrian's full trajectory. Luckily, the multi-view aspect of the datasets Crossroads supports allows users to recover the complete trajectory information. By clicking on the corresponding bounding boxes representing the same pedestrian in different views using the mosaic component, described in Section 4.5, Crossroads uses accurate pose information to reconstruct the full trajectory. In the case presented in Figure 4.3, after manually reconstructing the pedestrian trajectory, it gets easier to see the pedestrian stopped in the middle of the crosswalk while checking for any vehicles coming in their direction.

Finally, visualizing pedestrians' gait can uncover important patterns, potentially highlighting people at low mobility levels. The process described above also supports our system in improving the 3D pose of pedestrians, uncovering more accurate gait information. As presented in Figure 4.3, a comparison between the automatic pose reconstruction and the augmented reconstruction is supported by user annotation. Although the trajectory metrics (speed, acceleration, and change in direction) only suffer from minimum changes, the gait information can help us characterize the pedestrians. For instance, double support (DS) time is the time spent with both feet on the ground during the gait cycle. Both feet on the ground allow for greater stability and control of direction [167]. Higher DS time and larger variability in DS time are associated with higher risks of falling [97, 246]. Gait analysis of pedestrian allows for urban planners to prepare and adapt the current infrastructure to benefit people of all motor skills.

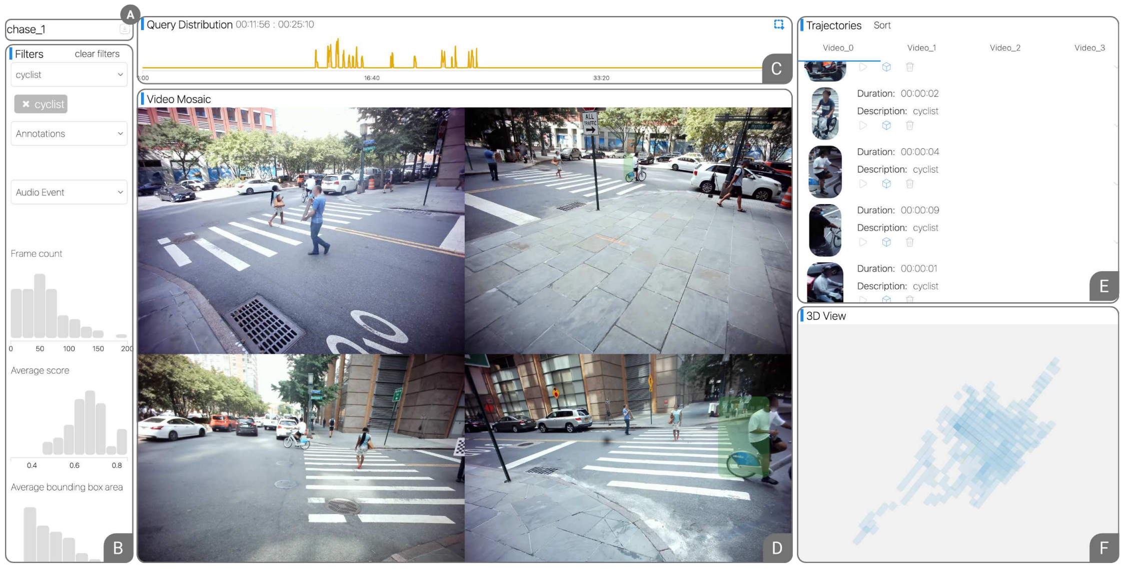


Figure 4.5: The Crossroads' Crossing Analysis view supports users in exploring the crossing data with different visual metaphors and query widgets. (A) allows users to look at global statistics of different locations and load data for a selected session. (B)) shows filters to select trajectories that follow a desired criterion. (C) shows the density of events that match the query with the selected filters. (D) shows a video mosaic of all the views available for the intersection. (E) display the selected trajectories in a list fashion, showing detailed information about them. Lastly, (F) shows a heat map of cyclist density at the intersection.

4.5.3 Interface & Interactions

In this section, we describe the key components of the Crossroads interface, which enable users to analyze urban crossings using the StreetAware dataset. We guide the development of our interface by well-established information visualization principles, where users can interact with visual metaphors summarizing different aspects of the data while providing mechanisms to highlight fine-grained details of the data. Crossroads implements two main views. The Registration Helper assists users in annotating spatial features from the videos to facilitate camera registration. The analysis view provides query tools and visual metaphors to summarize the data, integrating 2D, 3D, and audio information.

Registration Helper As described in Section 4.3, our tool must enable the exploration of automatically extracted physical measurements of actors, such as pedestrians, cyclists, and cars interacting within the crossing area (requirement **R2**). These measurements are based on dynamic 3D representations. However, conventional techniques for recovering this 3D information require camera registration techniques that are prone to failure in the StreetAware dataset's camera setup.

We overcome this challenge by introducing the Registration Helper view, which enables users to define matching points manually across videos from different cameras. These matching points will allow us to register the cameras with a few annotated points accurately. In this view, besides defining matching points, the user can also define shapes that will support the analysis of the intersections in the 3D visualizations provided in the Analysis view. This view consists of three key components. First, a mosaic of frames is generated based on the user's selection of a specific frame number, as shown in Figure 4.4. Once the mosaic is displayed, users can define matching points by clicking on corresponding points in each image. A lens-inspired widget, which zooms into the image region under the mouse pointer, assists in precisely annotating these matching points. Once matching points are set, users can define 3D shapes based on these points. The system automatically generates the 3D shapes and renders them in the other views to support the analysis of crossing regions using the user-defined points. This feature is especially useful for delineating sidewalk and crosswalk areas in the visual representations, providing essential spatial context for the analyses conducted by system users.

Analysis View At the core of Crossroads, the Analysis view (shown in Figure 4.5) enables users to identify common patterns and potential outlier events by combining query widgets for filtering trajectories of interest (supporting **R4**) with customized visual metaphors to summarize and explore details of the selected trajectories (supporting **R5**). Moreover, it also supports queries across multimodal data (supporting **R1**), enabling users to correlate audio events with pedestrian behavior. The interaction in this view starts with the user clicking on the session selector to pick a session of interest. The session selector button (Figure 4.5-A) opens a modal where a list of available session recordings is presented. An aerial image of the region is displayed for each recorded session in the database. Also, the users can click on the map view to explore where the recording session took place on Google Maps. Once a session is selected and the data is loaded into the view, populating the filters list (Figure 4.5-B). The filters combine drop-down lists with histograms showing the distribution of metrics relative to each trajectory in the session. For example, users might be interested in analyzing events where cars and cyclists share the intersection region. So, by selecting the corresponding classes using the class drop-down, all the other filters will be updated, showing the distribution of metrics for the selected classes, such as the distribution of speeds. Moreover, users can combine filters to build more complex queries. By brushing the histograms on the filters list, it is possible to specify fast pedestrians walking through the intersection while the traffic light is red.

Once the filters are updated, the temporal distribution component highlights when trajectories satisfying the applied filters are present by showing area charts with different colors, representing different classes (Figure 4.5-C). This feature supports the identification of times when selected classes coexist in the intersection region. The user then interacts with this visual representation to brush time ranges and load the individual trajectories to the trajectory list component and the 3D view (Figure 4.5-E and F), respectively.

On the trajectory list, all the selected trajectories are shown with detailed information, such as the duration it is visible on the video and the class name.

Users can expand each entry in the list to get more information, such as speed, change in velocity, traffic light information, and audio event occurrences. Users can click on the play icon to play the video segments when the trajectory is visible in the mosaic view (Figure 4.5-D). In the mosaic view, the bounding boxes of every trajectory happening at the same time interval are overlaid in each of the videos. This information supports users in matching trajectories across videos to generate detailed 3D pose information, allowing for the gait analysis of pedestrians and noise removal from automatically generated trajectories. This process is further explored in Section 4.5.2.

Lastly, the reconstruction view (Figure 4.5-F) enables the trajectory comparison of different video segments and summarization of the current selection. The user can toggle between a heatmap representation and a trajectory representation. The heatmap summarizes the average occupancy of the intersection, while the trajectories representation allows the user to compare different trajectories. Users can also click on specific trajectories to load detailed pose information. Clicking on trajectories also triggers the mosaic component to play the corresponding video. This view of the trajectories enables the comparison of events happening at different moments of the video (requirement **R4**).

4.6 Usage Scenarios

To evaluate Crossroads, we detail two usage scenarios created in collaboration with an urban planner with over 10 years of experience. These scenarios demonstrate the key features of Crossroads and illustrate how the system can be applied to address real-world challenges. The scenarios were subsequently reviewed by two additional experts, whose feedback is discussed in Section 4.7.

4.6.1 Global Analysis of Intersections

To facilitate high-level exploration and comparison across multiple urban intersections, our tool includes global summaries of the recording sessions (Figure 4.6-A). Each session is represented by a unique name and a snapshot

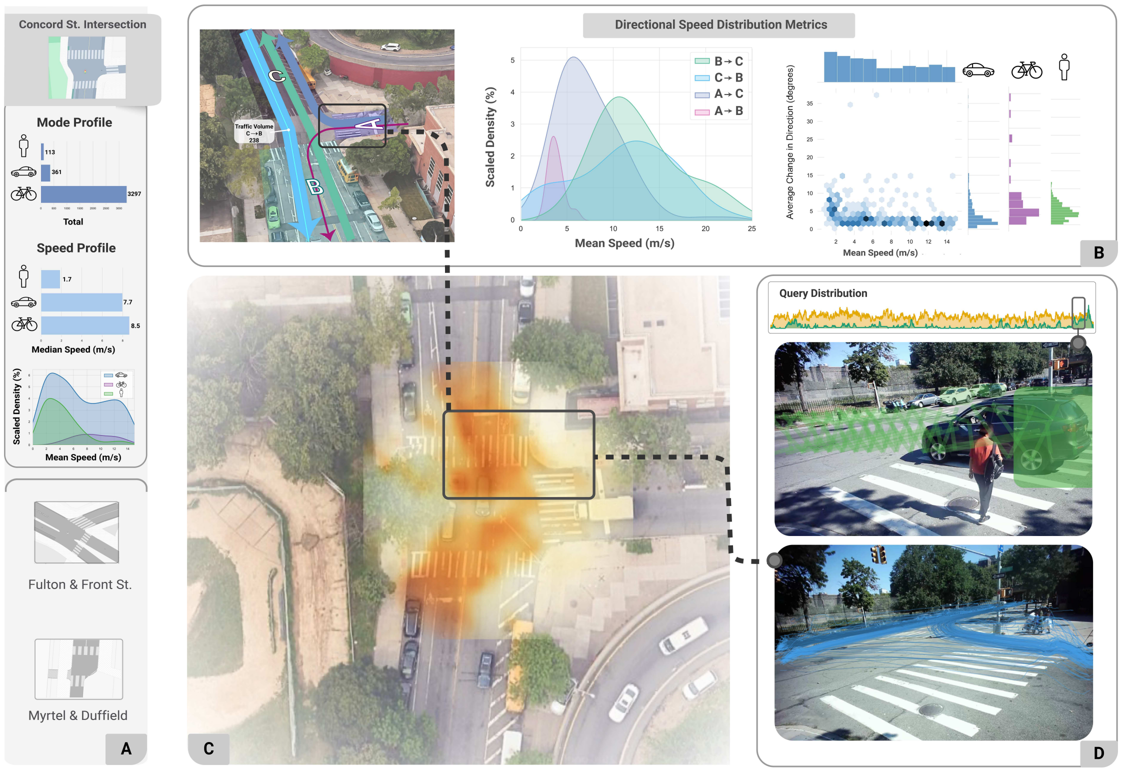


Figure 4.6: Crossroads, an interactive visual analytics tool for exploring multimodal movement and safety patterns at urban intersections. (A) provides a global overview, where users select from a list of intersections and view summary statistics including agent counts and speed distributions. (B) presents directionally disaggregated flow metrics and a hexplot that captures behavior signatures by plotting mean speed against directional change across modes. (C) overlays a speed heatmap onto a satellite image, highlighting localized high-speed segments—such as fast right turns, often associated with elevated pedestrian risk. (D) enables semantic querying within selected regions to retrieve frames where vehicles and pedestrians co-occur, revealing potentially hazardous interactions such as near-miss incidents. Together, these panels support a multiscale, multimodal workflow from high-level pattern recognition to detailed, evidence-based safety investigation.

of its location, enabling users to quickly situate the site geographically. Expanding the session card, a detailed summary is presented, where users can inspect mode-specific activity and speed characteristics. The mode profile includes counts and types of observed agents—pedestrians, cyclists, and vehicles—captured across all cameras. The speed profile shows median and mean moving speeds per mode. These offer a concise statistical fingerprint of each intersection to guide deeper investigation.

At our Concord intersection, one of the locations available in StreetAware, the mode profile reveals a disproportionately high volume of vehicular traffic relative to pedestrian activity, suggesting a vehicle-dominant intersection with limited pedestrian flow. The speed profile further reveals that vehicular speeds are not only higher in absolute terms, but also display lower variability, indicating a consistent flow pattern—often characteristic of intersections with minimal stopping or yielding.

By computing trajectories using our system, we inspect directional movement patterns (Figure 4.6-B). By doing so, a clear asymmetry emerges: the A→C right-turn leg exhibits both high traffic volume (281 vehicles) and elevated speeds relative to other directions. The kernel density estimate plot shows the speed distribution for this leg skewed higher than others, and the KDE peak for A→C surpasses 10 m/s. Compared with the average vehicle speeds across all intersections in the dataset, approximately 6.1 m/s based on the heatmap summary, shown in the same section. This pattern is further visualized in Figure 4.6-C, highlighting a spatial concentration of high-speed movement along the right-turn path. This path intersects directly with the pedestrian crosswalk, increasing the risk of conflict.

Figure 4.6-D enables further inspection through semantic querying: selecting the crosswalk region surfaces trajectories where vehicles and pedestrians co-occur. One such frame reveals a near-miss scenario—a pedestrian jaywalking is caught in the intersection while a vehicle proceeds with a high-speed right turn.

This case exemplifies how Concord St.'s turning behavior deviates from normative patterns, and how our system guides users from statistical anomalies to real-world behavioral evidence. It underscores the value of integrating direc-

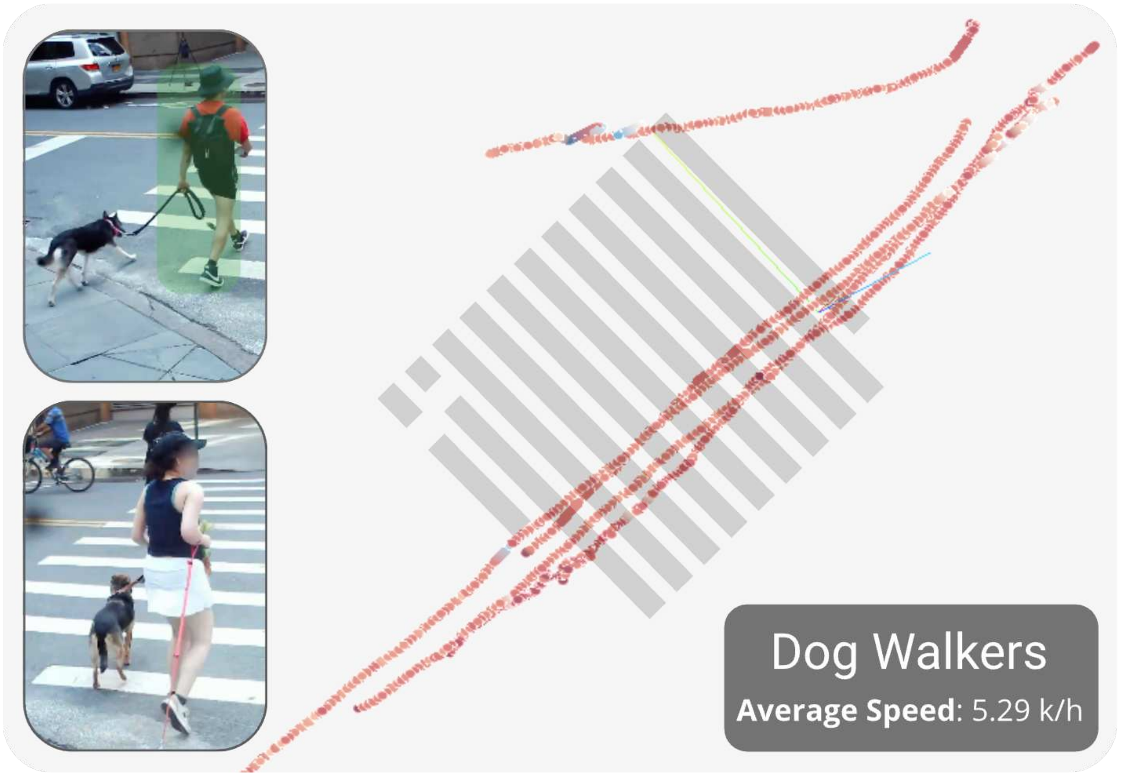


Figure 4.7: Detailed representation of pedestrians walking with dogs. Crossroads summarizes trajectories and metrics of different actors using the crossing region.

tional speed, volume, and spatial-semantic queries into intersection analysis workflows.

4.6.2 Analysis of Crossing Patterns for Design Intervention

Urban crossings are critical points where diverse groups of pedestrians interact with vehicular traffic, creating complex movement patterns that require thoughtful design interventions. Vulnerable groups, such as guardians with young children, people pushing strollers, and dog walkers, often exhibit sporadic and less predictable trajectories that standard design measures fail to accommodate. Crossroads can facilitate discovering these crossing behaviors and inform targeted design interventions. This scenario demonstrates how the tool enables a traffic researcher to: 1) identify high-risk pedestrian groups, 2) analyze distinct behavioral patterns, and 3) propose design interventions based on the observed patterns by analyzing videos from intersections with varying densities.

Identifying vulnerable pedestrian groups The analysis begins by identifying high-risk groups, specifically parents with children and individuals walking with dogs, noted for their unpredictable trajectories in transportation research [144]. Using the Analysis view, as presented in Figure 4.5, the user filters the trajectory list to locate specific targets, such as strollers and dogs. The user starts by analyzing patterns of pedestrians walking dogs. After filtering trajectories of dogs and pedestrians, the temporal distribution component updates, showing times when dogs and pedestrians are sharing the street. The same process is done for people carrying strollers. The user then scrolls over the trajectory list to find the trajectories corresponding to the pedestrians walking with dogs. Then, the user annotates all the dog walkers to allow for the comparison of their trajectories and speeds using the reconstruction view. Lastly, the user also identifies some peaks in the number of people crossing at specific times, suggesting a crowd movement in the crossing region.

Behavioral profiling and speed analysis The next step involves analyzing the identified groups' crossing patterns and speeds. The user reviews the video instances annotated in the analysis view to observe pedestrian groups under different conditions, such as crowded versus normal traffic flows. In crowded settings, pedestrian speed is often constrained by the group's lead, indicating a need for distinct signal timing strategies in high-density scenarios. Using Crossroads empowers the user to analyze micro-behaviors, subtle, moment-to-moment actions taken by pedestrians, which play a crucial role in understanding risk assessment and adaptive responses at urban intersections. These behaviors include hesitations, stops, speed changes, and interactions with environmental cues like sounds or nearby vehicles. Capturing these micro-behaviors provides invaluable insights into how pedestrians, particularly vulnerable users, assess risks and make real-time decisions at crossings. Behavioral profiling shows that people walking with children often struggle to keep them within the intersection boundaries, resulting in deviations from a straight path and an average crossing speed of 0.8 m/s. Front carrying an infant or a child can reduce the average speed by around 10%. Those pushing strollers display more cautious behavior while crossing with an average speed of 1.24 m/s, often stopping mid-crossing. Finally, as shown in Figure 4.7, pedestrians walking with dogs display a similar behavior, showing a straight trajectory to cross the street at an average speed of 1.46 m/s. The reconstruction view makes it easier to detect a case where a dog walker leaves the crosswalk region to reach their destination quickly.

Propose design interventions Based on the observed behaviors, key interventions include: 1) widening crosswalks in crowded intersections to handle larger pedestrian volumes, 2) increasing the size of intersection box, and widening the space between the stop line and crossing line to provide a buffer for unpredictable movements, and 3) implementing adaptive signal timing based on real-time crowd density and pedestrian group dynamics to ensure safe crossing for vulnerable groups.

Throughout the analysis, Crossroads 's interactive 3D reconstruction, object detection, dynamic filtering, and video review, enhance the accuracy and applicability of the analysis, ensuring that proposed interventions are directly grounded in observed pedestrian behavior. This scenario highlights how Crossroads bridges the gap between micro-behavior analysis and tactical urban design, offering insights not possible using top-down view analysis and supporting the design of safer, more inclusive intersections.

4.6.3 Uncovering High-Risk Patterns with Multimodal Queries

Distracted pedestrian behaviors pose serious safety risks and often lead to near-miss incidents. High pedestrian volumes combined with inattention create complex and hazardous crossing scenarios that standard safety measures frequently fail to address. Consequently, these situations are likely to produce interactions between drivers and pedestrians represented in audio events, such as car honks. Moreover, pedestrian might change their behavior when ambulances or

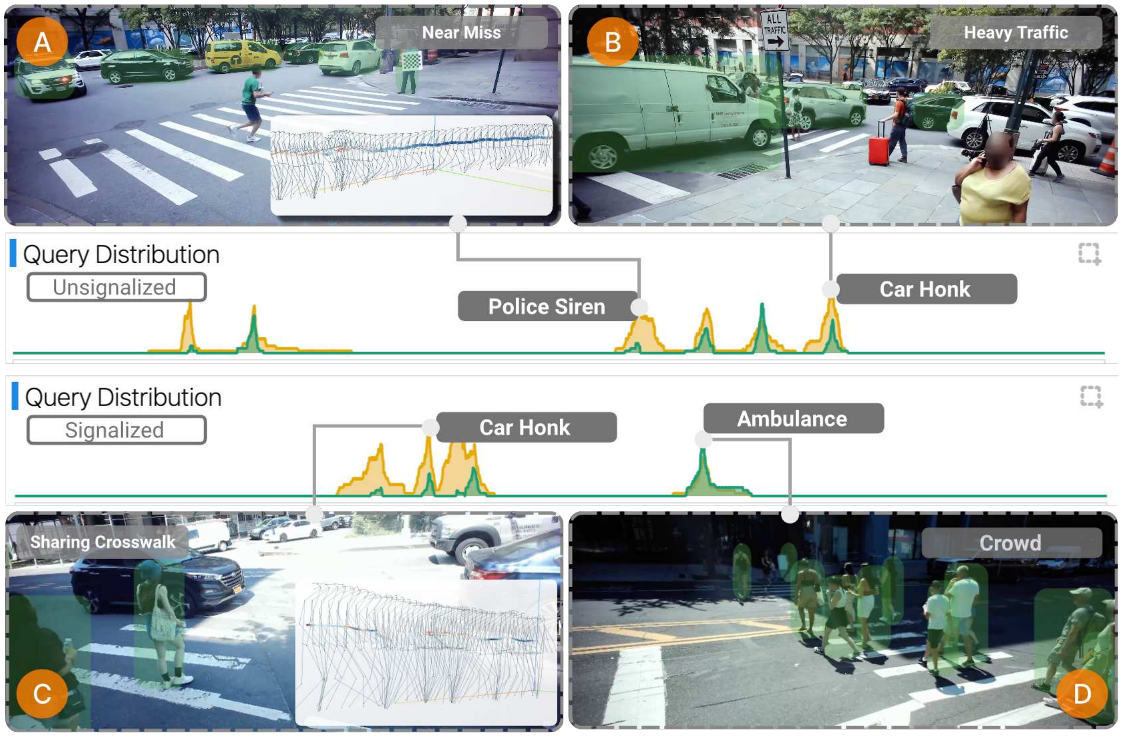


Figure 4.8: Audio events and correlated high-risk scenarios. (A) shows a pedestrian accelerating due to a police car approaching the intersection. (B) car honks due to heavy traffic. (C) jaywalker sharing the crosswalk with a car. (D) crowd crossing a lower speed.

police cars are approaching the intersection. Here, we use Crossroads's ability to query for audio events together with pedestrian speed and acceleration to find potential high-risk scenarios and common driver behavior under different circumstances. The insights gained from this analysis are crucial for designing interventions that mitigate these common yet dangerous pedestrian actions.

Motivated by findings from the Federal Highway Administration's study [3], which highlighted the increased frequency of high-risk and near-miss scenarios at unsignalized intersections, the user plans to compare two intersections. The process begins with the user loading session data of a signalized intersection in Crossroads. Two filters are then applied in the filters list. First, the user filters for car honks and police sirens audio events, and reduces to scope to consider only trajectories of pedestrians and cars. Once these filters are applied, the query distribution component shows moments where the number of cars is similar to the number of pedestrians and moments where the number of cars is much higher than the number of pedestrians. The user continues to inspect the configuration of both cases by selecting and replaying the video segment of each case. The mosaic views show that when fewer pedestrians are present in the intersection, the car honk is usually motivated by heavy traffic, as represented in Figure 4.8. However, when inspecting one of the periods when a police siren sound was detected, one of the trajectories shown on the 3D view presents a significant, abrupt change in speed. By clicking on it, both the mosaic viewer plays it and the reconstructed 3D poses appear on the screen. The user then sees a police car turning onto the intersection while the pedestrian accelerates to cross. This behavior suggests that part of the intersection might be occluded with heavy traffic, and this lower visibility poses a higher risk of accidents. An intervention might be needed to alert pedestrians that cars can turn into the crossing region at high speeds or to force drivers to reduce speed when near the crosswalk.

The exploration process moves to the signalized intersection. The user repeats the same process to filter for trajectories when there is a car honk detected and update the query distribution view. A similar pattern is shown. However, most car honks at this intersection are now related to drivers trying to make other cars move faster at the traffic light change. Then, moving the attention towards the reconstruction view, the user detects a moment when a car and a pedestrian share the crosswalk at the green light, suggesting a jaywalking event. The user plays the trajectory in the mosaic view and confirms the contravention: the pedestrian waits on the street instead of the sidewalk, looking for a halt in traffic flow, and, once it happens, takes advantage to quickly cross the street while the traffic light is still green for vehicles. This can turn dangerous quickly when larger vehicles, such as the SUV and pick-up truck seen in the video, are at the intersection. Although it was not the case for the jaywalking event being analyzed, these larger vehicles reduce the visibility of pedestrians crossing the street, and it can become particularly dangerous with cyclists moving fast through these patches of low visibility while a pedestrian jaywalks. This behavior motivates a more in-depth study of traffic light cycles in this region, giving more time for pedestrians to cross.

4.7 Experts' Feedback

To further evaluate Crossroads, we introduced the system and usage scenarios to two practitioners with Ph.D. degrees in transportation engineering to use the tool and reflect on their experiences. One practitioner is a transportation modeler working in the public sector for the city of Boston, and the other is a traffic engineer working in the private sector in Sacramento. Both provided valuable insights into the usability and effectiveness of Crossroads, highlighting its strengths and suggesting areas for improvement.

4.7.1 Feedback from the Transportation Modeler

The transportation modeler found Crossroads 's object detection and interactive features particularly valuable, though she noted challenges with spatial orientation within the tool. Key suggestions include:

Effectiveness of the interactive feedback loop and object detection features She praised the object detection capabilities for accurately identifying various agents, including pedestrians, cyclists, dogs, and strollers. The correction feature allowed precise adjustments to mismatched data, enhancing the reliability of trajectory information. She emphasized, *"The combination of these features sets this tool apart, as it avoids over-reliance on black-box algorithms by enabling expert intervention in sensitive contexts that directly impact public safety."* This hands-on approach aligned with her preference for transparent, expert-driven analysis, especially in areas affecting public well-being.

Impact of advanced filtering options The practitioner was impressed by the advanced filtering options, particularly the ability to combine height and gait patterns with filters such as pedestrian light flicker. She noted, *"These capabilities significantly bolster the analysis of pedestrian behavior and safety assessments at intersections, providing valuable insights for transportation planning and safety analysis."*

Recommendations for orientation enhancements Despite the tool's strengths, she experienced disorientation within the map viewer and suggested several enhancements: incorporating a north arrow to help users orient themselves, a reset button to return to a default viewing angle, and a base map overlay to provide contextual information. She emphasized that these additions would enhance situational awareness and help link trajectories with the real-world features of the selected intersection.

4.7.2 Feedback from the Traffic Engineer

The traffic engineer, working in the private sector in Sacramento, frequently analyzes large multimodal trajectory datasets and highlighted the value of Crossroads 's 3D visualizations and advanced spatial filtering capabilities. Key suggestions and feedback include:

Insights from 3D trajectory visualization and aggregated metrics He noted that *"the 3D visualization of different modes of movement (pedestrians, cyclists, and vehicles), can provide detailed insights at a finer scale while still allowing me to assess aggregated metrics such as traffic volume on each branch or turning cyclist volumes."* He appreciated the spatial filtering options, which enabled him to query for potential conflicts at high-turning volume areas, facilitating a comprehensive conflict analysis.

Integration of noise data for risk assessment The expert also found the integration of noise data, particularly honking sounds, to be valuable for identifying high-risk or alert scenarios. He emphasized that *"this feature could uncover moments of heightened stress or danger that would otherwise be missed in traditional traffic analysis."*

Suggestions for enhanced orientation and comparative analysis To improve spatial awareness, he recommended adding base maps—either imagery or vector-based—to contextualize the scene better. He also suggested enabling side-by-side views of multiple intersections to facilitate comparative analysis, allowing traffic engineers to assess and compare conditions across different urban settings more effectively.

4.7.3 Summary of Experts' Feedback

Both the public and private sector experts underscored the significant potential of Crossroads for enhancing intersection analysis. They highlighted the importance of object detection, advanced filtering, and spatial querying capabilities in delivering actionable insights. However, they also noted areas for improvement, particularly in user orientation within the 3D environment and enhancing the tool's capacity for comparative analysis across multiple intersections. These insights directly inform our ongoing development of Crossroads, ensuring it continues to meet the needs of practitioners in real-world settings by providing precise, transparent, and adaptable tools for urban safety and traffic planning.

4.8 Conclusion and Future Work

In this paper, we presented Crossroads, a system to enable the analysis of data representing intersection dynamics. Our approach provides effective analytical capabilities, minimizing reliance on black-box procedures in transportation and safety planning and enhancing transparency in analysis and decision-making. Combined with StreetAware data, this approach bolsters human-centered analysis and tactical urbanism, shifting away from top-down perspectives that often neglect vulnerable users, like mothers front-carrying children, or miss micro-behaviors linked to distress, interactions, or responses to environmental noise and stressors.

Looking forward, we plan to tackle some of the limitations of the current system version. We plan to do an in-depth error analysis of the reconstructed information and incorporate feedback collected from system users. Lastly, we envision this framework to be extensible to other types of human activity, such as humans practicing sports. While our evaluation approach finds parallels in previous studies [154], we recognize the need for a broader assessment. To this end, we plan to conduct a larger-scale evaluation involving experts from various cities, allowing us to capture a wider range of urban contexts and planning challenges.

Chapter 5

Urban Rhapsody: Large-scale exploration of urban soundscapes

5.1 Introduction

City soundscapes represent a rich source of information about urban systems, such as transportation, civil construction, and social activity. Low-cost sensors can be used to capture aspects of this acoustic environment, and computational methods for large-scale data analysis offer new approaches to characterizing the different contributing sources. Such understanding offers insight into how a city behaves through space and time (e.g., *"what are the typical sounds in a neighborhood during the night?"*), and can help in tackling various urban problems such as noise pollution. The research reported here was undertaken in partnership with researchers from one such sensing initiative, the Sounds of New York City (SONYC) project [18], who have developed and deployed low-cost sensors to measure and stream real-time sound pressure level (SPL) and audio data. To date, more than fifty sensors have been deployed throughout New York City (NYC), collecting data for over five years (in total, more than 60 TB). To meaningfully understand this data, the project's researchers are developing new machine listening models that 1) extract audio embeddings and 2) classify these sounds based on a set of predefined labels. However, these tasks pose several challenges that impede even state-of-the-art models' effectiveness in capturing the urban soundscape's dynamism. First, audio is complex, a recording typically captures different sound sources (e.g., dogs barking and people talking) simultaneously. Second, sound events are transient (e.g., a honking car horn) but in aggregation can last for hours (e.g., car engines on a busy highway). Third, audio has a temporal aspect, and so unlike images or words, sounds do not have a straightforward pictorial representation, limiting our ability to quickly review a large collection of recordings in parallel. Hundreds of images can be reviewed at the same time, with objects identified in minutes. However, looking for patterns or events in a large collection of audio data often requires listening to hours of individual recordings one after another. Analyzing audio data is time-consuming and hard to scale. This calls for novel techniques and visualization interfaces to facilitate the process, leveraging human expertise.

Motivated by these challenges and the need to gain new insights into the soundscape of the city, we introduce Urban Rhapsody, a framework for the interactive visual analysis of large collections of urban acoustic data. Using recent advances in machine listening to generate audio representations, Urban Rhapsody allows analysts to create a visual representation of the soundscape across different ranges of temporal and geographical granularity. We adopt a human-in-the-loop approach that enables users to interactively label data points, create new classification models based on their expertise of the soundscape, and assess the performance of audio classification tasks. Finally, because noise patterns might happen at different scales (minutes, days, months, etc.) in the urban environment, we employ a multilevel visualization scheme. Using case studies that demonstrate the utility of Urban Rhapsody, we showcase support for fast exploration of similar sounds or concepts, assessment of classification model outputs in different scenarios, geographical and temporal understanding of the embedding space, and summarization of soundscapes by key representative audio frames. Previous approaches to these challenges were either applied in a different context [57], or constrained to the analysis of sound pressure level (SPL) data [156], painting an incomplete picture regarding urban noise problems [271]. Urban Rhapsody is the first visual analytics framework that enables a comprehensive analysis of urban acoustic environments, going beyond time series to leverage a unique audio data set that enables a more comprehensive analysis. Our contributions can be summarized as follows: (1) A set of requirements, elicited in collaboration with SONYC's audio researchers, for visual exploration of large urban audio sets. (2) A set of visual interactions that enables users to iteratively construct audio machine learning models; (3) An interactive visual analysis framework, Urban Rhapsody, that supports concept-based exploration of large collections of audio recordings (such as the ones generated over the five-year deployment of the SONYC sensor network). We illustrate this with two case studies set in NYC, highlighting how our approach can be useful in tackling issues that have generated intense public debate. Our framework is also available on GitHub (`https://github.com/VIDA-NYU/Urban-Rhapsody`).

5.2 Background

According to the World Health Organization, in Western Europe alone, more than 1 million healthy life-years are lost annually to environmental noise pollution [180], and in NYC, an estimated 9 out of 10 adults are exposed to excessive noise levels [172]. This impacts public health [95], social well-being [93] and quality of life [64], as noise increases stress, sleep disruption, annoyance and distraction [29, 96, 162, 180]. To mitigate this, governments devise noise codes that typically consider SPL measurements in relation to time of the day/week and location and impose regulations that aim at mitigating the noise at the source (e.g., by erecting sound barriers around major roads or modifying building designs) [2, 30, 95]. However, enforcing these codes is time-consuming and costly, requiring trained inspectors to be present at sites to make assessments and capture sound carefully using calibrated equipment [18].

Beyond this, noise pollution can be highly subjective [56], and so quantitative SPL metrics may be insufficient [92, 195]. Because of this, there is a shift towards understanding the *source* of the noise, and to consider context in people's perception of sounds [194, 234]. Such a "soundscape approach" [31, 54, 184] views the acoustic environment as composed of both positive and negative sources [32]. Data gathered using SONYC sensors offers a unique opportunity to measure noise pollution quantitatively and additionally gain insights into the acoustic environment's qualitative characteristics. We can therefore conduct structured assessments at scale, accounting for both SPL and sound source. This raises important challenges (outlined in Section 5.4.2) that we seek to address in this research. Urban Rhapsody is the first step towards allowing domain experts to better understand the soundscape of complex cities such as NYC.

5.3 Related Work

5.3.1 Urban visual analytics

Urban areas are a major source of data that have tremendous potentials to improve policy making, enhance the lives of citizens, and pursue sustainable development. Visualization systems have for long been an important tool for the analysis of urban data [62, 272]. Several approaches use urban data to study different properties of a city, such as air pollution [270], public utility service problems [263], sunlight access [150], land use [190], human movement patterns [122, 151, 178], transportation [7, 75, 105, 241, 260], and also the relationship between these data sets [46, 60, 138]. More general tools, such as ArcGIS [110], Urbane [63, 74], and Vis-A-Ware [182] have facilitated the use of multiple urban data sets to help inform urban planning and decision making process.

In our previous work, Time Lattice [156], we have tackled the problem of noise pollution by proposing a data structure and visual interface that allowed experts to explore a large data set composed of SPL dB measurements from SONYC sensors. We only used SPL measurements without considering that the soundscape of a city is composed of different sources and can be perceived differently by different people. With Urban Rhapsody, our goal is to account for the user's knowledge and perception in the exploratory process of large collections of urban sound recordings in a vocabulary-free approach, meaning that users are free to explore the soundscape according to any concept they create. To the best of our knowledge, Urban Rhapsody is the first visual analytics system specifically designed to allow domain experts to explore a large collection of sound recordings of an urban environment.

5.3.2 Environmental sound representation

In recent years, several large audio data sets have been released that have moved the field of environmental machine listening forward [77, 86]. However, many audio classification tasks do not map onto the class vocabulary of these data sets and thus require additional labeling, which is time-consuming and costly. To address this problem, machine listening practitioners have turned to transfer learning [255] in recent years, which has been shown to be effective for many audio classification tasks [8, 11, 37, 50, 91, 107, 120, 225]. In transfer learning, models are typically pre-trained on large data sets using supervised [11, 101] or self-supervised learning [8, 37, 107, 120, 225], and the knowledge acquired during pre-training is re-used for tasks where data is limited. A common method of re-using this knowledge is to treat the pre-trained models as feature extractors, utilizing learned latent representations (i.e., embeddings) from within the pre-trained models as the inputs to models with little or no labeled data. Look, Listen, and Learn [8] is one such pre-trained model whose embeddings were shown to be discriminative in several environmental audio classification tasks [8, 37, 50, 245]. This model is pre-trained using self-supervision on an auxiliary task of audio-visual correspondence. In this work, we use OpenL3 [50], an open-source code of Look, Listen, and Learn, as an audio feature extractor to transform each audio recording into a series of embedding representations.

5.3.3 Machine-learning-aided multimedia exploration

Machine learning has opened a new horizon in data exploration across various fields, with numerous systems making use of the powerful capabilities it provides. For instance, Urban Mosaic [155] uses deep learning representations to search for patterns in a large collection of street-level images. II-20 [258] allows users to generate image classifiers using novel interactions. Previous works tried to explore the semantic meaning of the features extracted by deep learning models, as they do not always map into human-understandable semantic features: Embedding Projector [215] and Latent Space Cartography [131] enable the analysis of embedding spaces for multimedia data through multidimensional projections [111, 142] to enable users to understand features that might be encoded in the latent representations.

To create classification models that can recognize human-understandable features in multimedia data sets, previous approaches employ active learning frameworks leveraging the users as oracle annotators to annotate new samples the system identified as the most informative. For example, previous studies investigated the usefulness of active learning for labeling tasks [21] and its application in other fields such as anomaly detection [124, 126], commuting flow estimation [257], and image categorization [258]. These approaches often guide the user on choosing the next subsets of the data to label next to improve the performance of the model, which, in our case, can limit the user in applying their previous understanding of the soundscape to label the concepts [164]. Our proposal leverages a set of techniques presented in previous works to enable users to better understand the spatiotemporal distribution of events in acoustic recordings while accounting for their knowledge of the soundscape to build concept-based classification models to gain insights into the dynamics of the urban environment.

5.4 Sounds of New York City

The research reported in this paper was undertaken in conjunction with audio and machine listening experts from the SONYC project [18], and utilizes data generated by the project's sensors. Our collaborators have background in urban science and machine listening [37, 63, 155, 238]. In addition, the project communicates their findings to the media [33] and works closely with the NYC Department of Environmental Protection to understand their needs and investigate new ways of monitoring and mitigating noise pollution.

5.4.1 A data set of urban sounds

The SONYC acoustic sensor network consists of more than 50 sensors deployed around three boroughs of NYC: Manhattan, Brooklyn, and Queens. Figure 1.1 shows the spatial distribution of the sensors. These sensors are positioned 15 to 25 feet above the ground. To maintain the privacy of bystanders and prevent the recording of intelligible conversations, the sensors do not record continuously, but rather record 3 10-second recordings at random intervals within each minute of a day (i.e., for a single day, each sensor will record 720 minutes worth of 10-second audio recordings uniformly distributed throughout the 1440 minutes of the day). As of 2021, SONYC has collected approximately 1,700,000 hours of SPL data (stored as second or millisecond resolution timeseries), and 877,000 hours of recorded audio. To extract a discriminative, lower-dimensional representation of each 10-second recording, we employ OpenL3 trained on an environmental sound subset of AudioSet [50]. OpenL3 is an open-source library for computing deep audio embeddings, developed by researchers from SONYC, and its design choices were informed by the need to classify sounds from urban environments. For each 10-second recording, we use a hop size and window size of 1.0 second (with centered windows) to compute 10 512-dimensional feature vectors. This produces a feature vector that coarsely captures the *general* acoustic aspects of the environment.

5.4.2 Challenges

The complexity of the urban environment brings several challenges when it comes to analyzing and extracting insights from urban sound data, especially considering such a large data set as the one captured by SONYC.

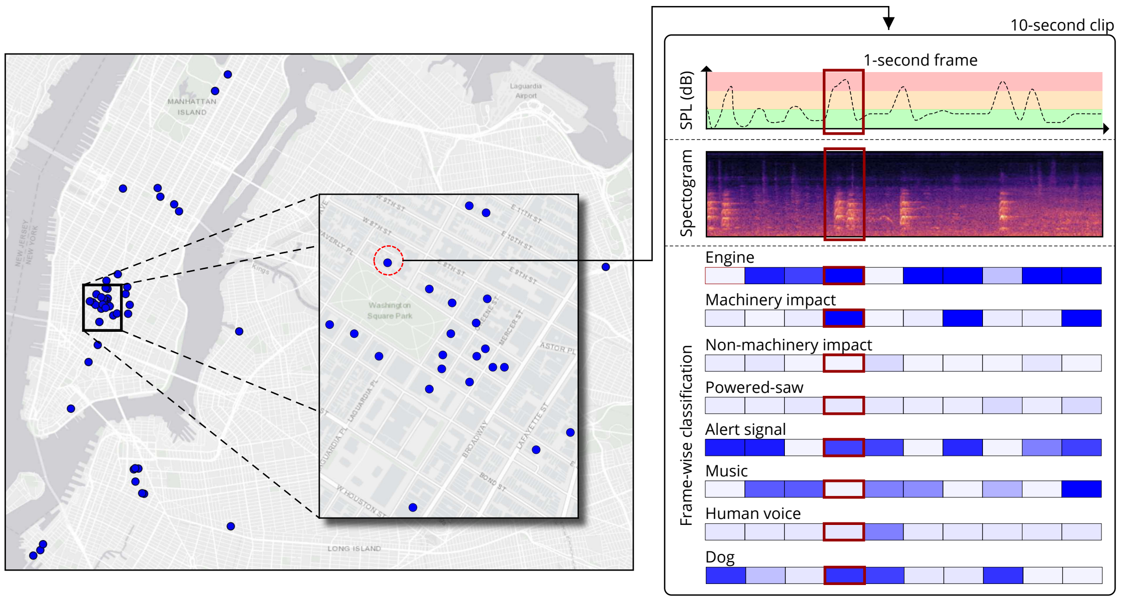


Figure 5.1: Spatial distribution of SONYC sensors (left) showing the coverage of the city. Right image illustrates the data from a sensor located near a park in Manhattan. Sensors record both the sound pressure level at each second (SPL dB), as well as the environmental sounds (stored as 10-second clips). For each 1-second frame in the clip (highlighted in red), we compute the classification considering user-crated prototypes. The figure shows classes following standard urban audio taxonomies.

Sound representation. In complex environments such as cities, many sound classes seem quite similar, such as car alarms and sirens, but are distinct in the noise code and should be treated as such. On top of that, when handling sounds from cities, the acoustic environment changes by location and by time within seasonal cycles. As a self-supervised method, OpenL3 does not need human-generated labels to be trained, while still providing good sensitivity to different urban sounds. However, it falls short of properly accounting for *all* of the complexity of the soundscape of a city.

Mixture of sounds. Unlike images, where visual objects are opaque, sound objects are conceptually *transparent*, meaning that multiple objects (sound sources) can have energy at the same frequency [248]. This is especially true in an environment as complex as cities, where sounds are emitted from multiple sources, creating a soundscape that, albeit quite characteristic, is very difficult to parse and understand. In other words, in a city, at any given instant in time, a sound recording might have a mixture of background (e.g., bird songs, dog barks) and foreground sounds (e.g., engine, party, sirens).

Sound exploration. Again unlike images, there is no clear pictorial representation of audio data. This gap between audio data and visual representation is challenging when building visual analytics systems. Visual objects are *opaque* (a given pixel in a visual image corresponds to only one object), whereas sound objects are *transparent* (multiple objects can have energy at the same frequency). Sounds are therefore serial objects: when assessing an image, we can visually *scan* it to identify each visual object in the scene, creating a visual map of the objects that can help us fully understand the scene. Sounds only exist at one moment in time; once the moment is gone, the sound is also gone. In other words, a user can only observe a sound one moment at a time, unlike images where we can observe multiple objects at a time. In spectrograms representation, similar neighboring pixels cannot be assumed to belong to the same object (i.e., frequencies are non-locally distributed on the spectrogram [248]). As we can notice, creating visual representations of sounds is a challenge, specifically considering a scenario with multiple sound sources, such as urban soundscapes.

Sound labeling and classification. Although previously proposed classifiers provide a reasonable link between embeddings and human-understandable vocabulary, their class vocabularies are limited, providing a narrow view of the rich and varied soundscape of the city, which is comprised of numerous types of sound events. Furthermore, manually labeling sound data to be used as groundtruth for model training is a laborious process. As previously mentioned, sounds are serial objects where the user needs to listen to one at a time, limiting the number of audio files that can be labelled in a short period of time. Purely automated mechanisms, however, are prone to misclassifications given the complexity of soundscapes.

Data size. Over the past five years, SONYC has generated more than 60 TB of data, including high-resolution SPL timeseries and audio recordings. If we consider the embeddings computed with OpenL3, we have 86,400 feature vectors with size 512 (177 MB in total) *per sensor per day*. Any visualization system must properly handle such data size to be interactive [132], either by sampling, filtering or aggregating the data.

5.5 Requirements

In our collaboration with machine listening researchers, over the course of two years in the context of the SONYC project, we established a set of requirements for a visual analytics tool to facilitate their analysis workflows. We then validated the working system through interactive demo sessions exploring a number of potential use cases. Underlying our work is the necessity to account for user knowledge when exploring the urban soundscape for different concepts. During these meetings, we identified the following main tasks that the experts desire to perform with the tool: 1) Select and listen to sound recordings from a set of sensors, considering different days of the week and time ranges; 2) Considering a query audio, quickly identify a set of possible similar sounds throughout a long period; 3) Create and refine classification models that allow for searching of complex sound scenes; 4) Assess classification performance interactively. To accomplish the listed tasks, we identified the following system requirements:

[R1] Interactive identification and labeling of similar sounds. Given the highly complex acoustic environment we observe in cities, audio representations cannot encode specific audio events that users might be interested in. Moreover, the high-dimensional nature of audio representations makes it hard to visually analyze such data, making multidimensional projection techniques a standard in this process. However, in many cases, user-perceived similarities between sets of audio frames (i.e., a one-second slice of the ten-second audio snippet) might not be represented in the selected projection technique, e.g., similar frames are far apart in the projected space (low-dimensional space), making it harder for users to find similar audio frames. Hence, finding similar audio frames based on user's perception is one of the requirements of the Urban Rhapsody framework.

[R2] Projection steering based on user perception. When exploring audio embeddings extracted from urban recordings through multidimensional projections, we often recognize clusters that do not represent the user's perception of the soundscape. Based on the user's understanding of the data set expressed through labeled points, the system should provide the capability of producing new projections that better encode the user's perception.

[R3] **Iterative creation of classification models.** Considering that current machine listening models present certain limitations, the system should provide the capability to iteratively create new classification models based on the data points labeled by the user (and, therefore, the user's perception of the soundscape). The system should also support assessing the evolution of the model's convergence through successive iterations.

[R4] **Local and global sound perspectives.** Audio embeddings might possess certain characteristics that only become clear when analyzed locally or globally. Then, it is important for the user to assess their local characteristics and to relate one sound to its immediate neighborhood or distant clusters.

[R5] **Match between audio and visual representations.** Visualizing audio files in the frequency domain is important for the user when assessing the accuracy of both the embeddings and classifications. For instance, two sounds might have very similar spectro-temporal patterns and classifications but completely different embeddings; it is important, therefore, to further assess and create hypotheses on what led to these different outputs.

[R6] **Support interactive query times.** The system should support interactive queries to enable the easy and quick labeling of data points and the creation of classification models.

5.6 Urban Rhapsody

To satisfy the previous requirements, we introduce Urban Rhapsody. A visual analytics tool able to provide a human-centered exploration of the urban soundscape using prototypes created on the fly through different interaction mechanisms. Our description of the framework is broadly divided into three parts. First, we describe our approach to generate classification models (or *prototypes*) of different *concepts* denoting complex urban sound scenes. Second, we describe the different components of Urban Rhapsody's visual interface (also see accompanying video), followed by a discussion of its architecture and implementation.

5.6.1 Prototype-based interaction

In Urban Rhapsody, we would like to support the search for audio events based on concepts and not only based on a single audio event. Here, we use the term *concept* to refer to an abstract idea or a general representation of a category in mind, such as “crowded street”, which can be perceived differently by people. In one of our case studies, we describe a case where the user keeps refining their concept of construction while annotating new sounds that together compose the full picture of a construction. To allow for this kind of search, we define *prototypes*, a structure composed of a classification model and a set of representative audio frames that defines a user’s understanding of a concept.

The classification model learns how to distinguish between the audio frames that are part of a given concept according to the user’s perception represented by annotations made during the interaction process with the system. Once the user starts labeling a specific concept in Urban Rhapsody, they can generate a new classification model that will be trained using annotated frames as input. Since our goal is to find occurrences of specific concepts in our data set, we should train this model with a diverse enough sample of the data so it can generalize well to different scenarios. Given this constraint, we train our model to distinguish between two labels: positive (frame is part of the concept) and negative (not part of the concept). For positive labels, we use all the frames annotated as the concept we are interested in. For negative labels, we use frames explicitly annotated as not being part of a concept and a random sample of all frames in our data set twice as big as our set of positive-labeled frames. The classifier we train in Urban Rhapsody is based on the classic random forest algorithm using a standard parameter setting for audio classification [240]. However, any classification model capable of outputting a likelihood score of a data point belonging to a class can be used in Urban Rhapsody. In this version, the likelihood function is calculated as the average prediction score across the trees in the forest. This interaction supports requirement **R3**.

Following **R6**, Urban Rhapsody must be capable of providing interactive query times during the exploration process. However, the size of the data set handled by our framework blocks us from filtering interesting audio frames by scanning the entire data set and computing the prediction probability of a given model to generate our visualizations. For this reason, after every model refinement made by the user, we also calculate a set of representative audio frames that will help us sample the data set to a smaller size before filtering interesting points using the aforementioned classification model. We calculate representative points of a concept by selecting all the points annotated as being part of a concept by the user and running a density-based clustering algorithm on the positive-annotated frames for a concept. For each cluster, we calculate the frame closest to its centroid and add it to the set of representative frames of that concept. The representative audio frames also help the users keep track of the concept they are creating through their interaction with the system. We enable the user to use these representative points as query input for a concept search using an approximated nearest neighbors (ANN) query.

5.6.2 Visual interface

The visual interface was designed to provide the user with the ability to browse through the entire data set, identify and annotate concepts present in audio samples, and, finally, iteratively and interactively build prototype models that generalize these concepts over the entire data set. Figure 5.2 shows the different components of the visual interface. Next, we discuss the design of each visualization based on its functionality: provide easy navigation through the audio collection, enable the annotation of audio concepts, allow for the detailed inspection of individual audio samples and facilitate the evaluation of prototype models.

Audio collection navigation. The interface implements several strategies to enable navigating through our data set. The first is the Calendar View (Figure 5.2(a)). This component presents a calendar of the year with each cell representing a single day. Within each cell, we can visualize a bar chart representing the distribution of frames of a specific concept during the four

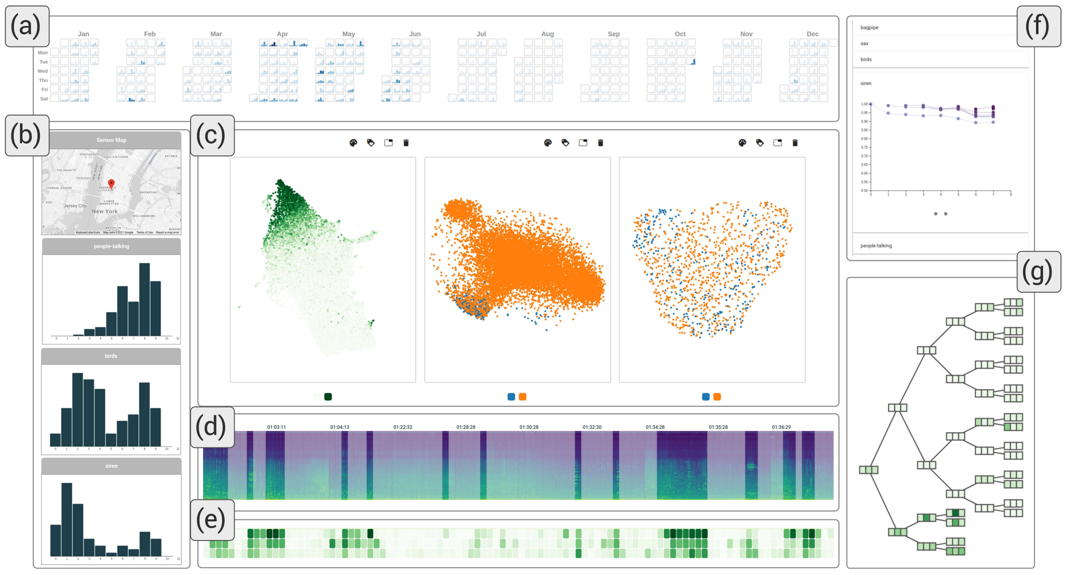


Figure 5.2: The Urban Rhapsody system visual interface: (a) Calendar View; (b) Sensor Map and Distribution View; (c) Day View (projections); (d) Focused View (spectrograms); (e) Frame Classification View; (f) Model Summary; (g) Mixture Explorer.

time slices of a day, allowing for the fast identification of the daily distribution of sounds. The bars of each cell are also colored according to the density of a specific concept in a day (more examples in a day will lead to darker blue bars). If a Calendar View cell is clicked, all the data corresponding to that specific day is loaded and in the day view (Figure 5.2(c)). Using the Day View, we can visualize the audio frames through the analysis of scatterplots generated by projecting high-dimensional feature vectors (audio embeddings) into a two-dimensional space using UMAP [142]. Although UMAP was the projection technique used for this version of Urban Rhapsody, given its dimensionality reduction capabilities, it is important to notice that Urban Rhapsody is agnostic of projection technique. The adaptation of the system to better accommodate experts' needs in terms of projection techniques is trivial. Here, the users can horizontally stack projections in three ways: reprojecting a subset of the data available for a day (i.e., reproject specific clusters to capture local structures of the data), removing a subset of the data, and reprojecting the remaining points (useful for removing clusters representing sensor failure, for example), and, lastly, steer the projections based on frames annotated by the user using a semi-supervised dimensionality reduction algorithm [224] that can learn a new low-dimensional space that better encodes the user's perception of the data (i.e., bringing frames with the same labels closer while keeping the different ones distant from each other), therefore supporting **R2**.

The projections in the Day View are linked and allow for selecting points through a bounding box or periods of the day. Selections update the Distribution View as well as the components designed for the individual inspection of audio samples, the Focused and the Frame Classification Views, shown in Figure 5.2(d, e) are described later in this section. At last, the projected points, each representing an audio frame, can be colored according to a likelihood of belonging to a concept or user annotation. When a day is loaded, Urban Rhapsody automatically calculates a hierarchical clustering of the points and updates the Mixture Explorer, represented in Figure 5.2(g) by a tree. Each node of the tree represents a cluster found by the algorithm. Each node is subdivided into subnodes, each being one concept that the user previously created. In the example presented in Figure 5.2(g) each subnode is representing a concept (people talking, birds, and siren from left to right) and is colored based on the average likelihood of the correspondent cluster contain the specific concept (darker green for higher likelihoods). If a node is clicked, the corresponding cluster is selected in the scatterplots and all the components of the interface are updated accordingly. For example, the node where all subnodes are darker green is where the user is more likely to find frames that contain all created concepts. It is important to notice that hierarchical clustering is a powerful visual strategy that enables the user to explore clusters of different sizes, both locally and globally (**R4**), and gain new insights into sound mixtures by focusing its inspection on cluster where previously created concepts are more likely to be found.

Annotation of audio concepts. One of the requirements elicited with domain experts is regarding the ability to annotate specific audio frames **(R1)**. To satisfy this requirement, Urban Rhapsody provides a mechanism to annotate specific audio frames that works as follows: users can select specific frames by using the selection mechanisms provided by the scatterplots or select a cluster using the hierarchical tree. Once a selection is made, the users can click on the labeling icon on top of the scatterplot to open a dialog that will allow for the annotation of these frames with as many labels as they want (positive labels). Also, users are able to annotate frames with negative labels, to explicitly say that a selection of frames is not part of a specific concept. This will help refine the prototype models when we find false positives during the exploration process.

Inspection of audio samples. To inspect details of an audio recording selected by the user during the exploration of the projections, Urban Rhapsody contains two widgets with visualization metaphors commonly used by audio experts: the Focused View (Figure 5.2(d)) and the Frame Classification View (Figure 5.2(e)). The Focused View shows a spectrogram of the audio samples selected in the projection. A spectrogram is a visual representation of the magnitude of the short-time Fourier transform, which describes the signal's energy by frequency as it varies with time. It can be visually encoded in a heatmatrix where each cell represents the intensity of a frequency in a given time. For example, the spectrogram of an audio file containing the sound of a siren contains wave patterns. Previous work investigated the usefulness of spectrograms in representing audio classes for humans and its performance in comparison to others standard audio visualizations [38]. We use this representation to allow the user to compare different sounds without having to listen to multiple audio files. The Frame Classification View displays the likelihood of observing a concept in the audio sample. In this way, the color of each cell of the matrix represents the probability of observing different sound classes in the associated audio frame. Finally, Urban Rhapsody allows the user to click on the spectrogram to listen to the recording. This interaction is important to bridge the gap between the visual representation and the actual audio (**R5**).

Evaluation of prototypes. As users keep creating and refining prototypes, they can evaluate its performance by making use of several components of our interface. First, for any given selection on the scatterplots, they can check a histogram showing the distribution of a concept's likelihood across the selected points. If the histogram is shifted to the right, it means the selection has a higher chance of belonging to a concept. Besides that, the users can assess the robustness of models in the Model Summary View (Figure 5.2(f)) where we present the evolution of the prototypes over the course of several refinements. Once we create a new version of a labeled subset for a specific concept, we train a new classification model to be part of the prototype and evaluate old versions of the prototype's classification model to assess the change in prediction over time. At some point, the user can come to a conclusion that labeling more points has no significant impact on the classification model and then stop the process.

5.6.3 Analysis flow

The exploration process starts with the user querying the data set using any of the three approaches we propose: select a frame from the examples we provide in a Query View as input for the similarity query, upload their own audio snippet and select a frame from this audio snippet, or query using one of the created prototypes. For all three query approaches, the user is able to select the number of frames the query will retrieve. Once the query is processed, the Calendar View is updated, showing the density of a given class, or concept, on each cell throughout the year (color), and its distribution within the day (bar chart). Next, the user can select a specific day and load all the available data for that day to further inspect the day's soundscape using the scatterplots in the Day View. At this point, the user can select specific regions of the scatterplot and listen to the correspondent audio frames, reproject specific regions of the day scatterplot to focus on local structures, remove undesired clusters or steer the scatterplot based on the annotation of frames. Also, color the points by prototype probability or created annotations. These operations will help users in two tasks: assess the performance of the prototypes they are creating and find data points that should be labeled as any concept of interest. Following that, it's possible to create different prototypes and refine existing ones based on new annotations the users are creating, either positive annotations or negative. Meanwhile, when prototypes are created and refined, the Model Summary gets updated, showing the change in prediction probability of the models and the set of representative frames of a given concept. When the user is confident about the prototype they are creating, they can reuse this prototype to query the entire data set and look for specific temporal patterns that a specific concept is happening. This analysis flow denotes the importance of having a user in the loop to evaluate the performance of the prototype models as Urban Rhapsody allows for the creation of concepts that match the user perception of the city's soundscape, which can not be evaluated quantitatively.

5.6.4 System implementation

We decided to develop Urban Rhapsody following a client-server architecture. We structured our application following microservices guidelines to ensure that we could effortlessly add new features to the tool and scale its deployment to make it available for the general public. The storage component keeps audio recordings and their embeddings located in different folders following the same naming convention for faster localization. Each audio file is also associated with a set of metadata attributes with temporal and spatial information (time of the recording and location of the sensor) that is kept in a separate database. The core of our application is composed of several microservices. The data server is responsible to serve audio files and spectrogram images. The web server provides users with a bundle of our Angular web application. The user server stores annotations on RocksDB [63]. The most complex services of our system are the ML server and the ANN server. The first is responsible for all machine-learning-related operations, such as multidimensional projections, hierarchical clustering, and model training. Following **R6**, the operations are processed using GPUs through RAPIDS libraries [197]. CPU-based libraries would not be able to handle such data-intensive operations required by Urban Rhapsody. The ANN server is responsible for computing similarity queries based on the euclidean distance between frames.

5.7 Case Studies

In this section, we demonstrate the application of Urban Rhapsody through two case studies using data from the SONYC sensors. In doing so, we highlight how the requirements listed in Section 5.5 are met in different tasks. The first case study explores how Urban Rhapsody can facilitate the interactive labeling and exploration of data for investigating out-of-hours construction noise, a pressing issue facing many large cities. The second one highlights another capability of Urban Rhapsody to facilitate searching for mixture of sounds to explore the impact of anthropogenic noises such as siren on bird songs. These case studies can be of interest to various stakeholders, from the general public and advocacy groups to government agencies, such as the Dept. of Environmental Protection.

5.7.1 After-hour construction noise

Construction noise is one of the primary sources of noise-related complaints in NYC. As the city grows, new structures are built, old ones get renovated, and economic pressures and deadlines lead developers to request the city for permits allowing them to perform construction outside the regular workday hours (i.e., 8 AM to 5 PM). In the past few years, this has been a major source of dispute between NYC residents and developers [140], and this problem is increasingly getting worse. In 2018, NYC's Department of Buildings issued around 67,000 after-hour permits, more than double the number of permits issued in 2012. Although developers must follow strict noise guidelines during after-hour constructions, the increase in the number of complaints related to these types of disturbances indicates otherwise. Even though the city constantly issues noise construction fines through manual inspections, the after-hour nature of these noises makes it especially hard to monitor them. This is a significant problem that needs to be addressed by cities and their different departments, with severe political, social, and economic ramifications.

In this study, we use the SONYC network to understand the impact of construction-related noises on the soundscape of NYC. Our first goal is to assess if these noises were captured by our sensors, to facilitate noise code enforcement activities. Secondly, we would like to use examples that we found during our initial exploration to build a prototype capable of pointing us to specific days and times where after-hour construction work might have

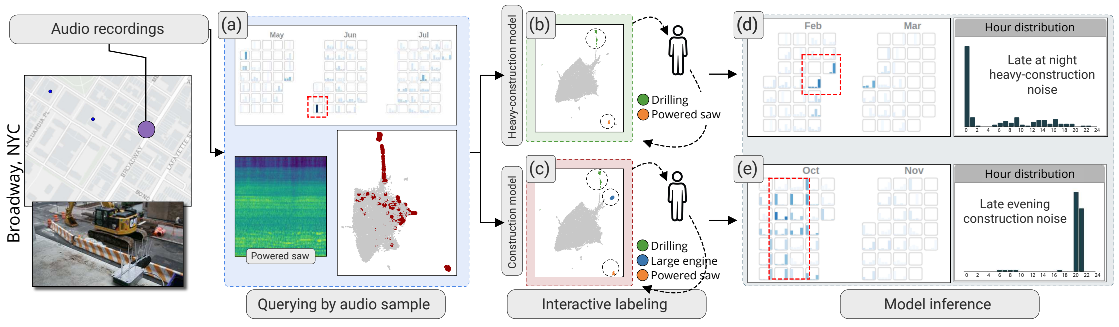


Figure 5.3: We use Urban Rhapsody to assess after-hour construction in New York City, first selecting audio recordings captured by sensors deployed around Broadway. Urban Rhapsody allows users to query using an audio sample, and drill down to days containing similar audios (a). Using the interactions provided by the tool, we are able to create classification models according to a user's perception of the soundscape (b,c), and then use these models to classify the entire data set and look for unusual events (d,e).

happened. We start by querying our data set for similar audio snippets using one of the examples provided in the system containing the recording of a powered saw **(R1)**. Using the Calendar View, we can quickly observe a day containing most of the similar audio excerpts according to our ANN model (Figure 5.3(a, top)). We select that day, and Urban Rhapsody generates a UMAP projection of all the audio frames within that day (Figure 5.3(a, bottom)). After a quick inspection of the projection scatterplot, we can notice a set of distinctive clusters (highlighted in red). Using the tool's interactions, we start by selecting the one cluster containing most of the points retrieved by the initial similarity query. By listening to a few recordings, we can notice that the points belonging to this cluster are perceptually similar to a powered saw, very common on construction sites **(R2)**. We also notice that most of these audio snippets were recorded around 8 AM, as the hour distribution chart shows us. Figure 5.3(b,c) highlights the recordings that happened around 8 AM, and it's possible to again see different clusters. After listening to recordings from each cluster, we noticed that each one of them represents different sounds (powered saw, drilling machine, engine). At this point, we can leverage Urban Rhapsody's feature that allows us to create models on the fly and decide whether to include certain sounds in our prototype **(R3)**.

Once we label recordings from that specific day, we generate two construction prototypes (with and without large engine noise). We can now use them to guide the exploration through different days of the year. This step allows us to speed up the search for similar sounds, without the need to listen to *hours and hours* of soundscape audio files. Also, during this guided exploration, we can adjust the prototype by labeling more points, either as negative or positive labels, as we assess the model's performance by listening to the recordings. This interactive process is highlighted in Figure 5.3(b,c), for two different models.

After refining our models once, we listened to the representative snippets of our prototypes and used them to look for unusual events. The calendar heatmaps show the results of the prototype queries (Figure 5.3(d,e)) where we can spot two interesting events. In February, we noticed that during two days, construction work happened during the night (Figure 5.3(d)). And that, during many days in October, the same engine noise started at 11 PM and lasted for approximately 30 minutes (Figure 5.3(e)).

To further validate this finding, we used citizen complaints filled through NYC's 311 non-emergency service phone number. Interestingly, there were actually a series of complaints reported on those two specific days of February. The ability to intuitively create prototypes based on audio files listened in the exploratory process sets Urban Rhapsody apart. Findings such as these not only highlight the usefulness of a *passive* network of sensors (as opposed to *active* sensors deployed in inspection visits), but also the usefulness of distinguishing different noises emitted from construction sites. Previous approaches, like Noise Profiler [156], focus on the SPL characteristics, a useful but crude measurement of noise. By enabling the exploration of specific types of noise, Urban Rhapsody can 1) provide a clearer picture of the soundscape near a construction site, 2) facilitate monitoring tasks carried out by enforcement agencies, and 3) validate the accuracy of 311 complaints.

5.7.2 Birds in New York City

The impacts of urban noise, air pollution, and the built environment on residents and migrating birds have been extensively studied [206]. There is

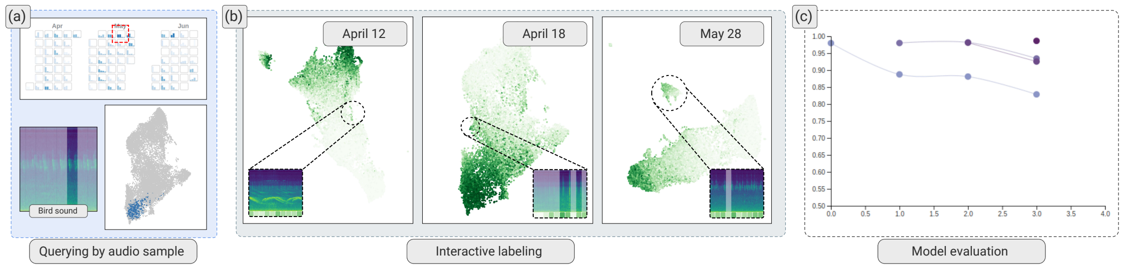


Figure 5.4: Interactive monitoring of the training process and refining the model. (a) We run a query using our sample birds' sound and analyze the clusters; (b) Investigating the clusters on different days to detect and re-label false positive and false negative instances, and refine the model; (c) The model evaluation indicates that our prototype models are converging as we do further iterations of refinement.

a strand of research that specifically analyze birdsong to discover if exposure to loud urban noise can lead to significant changes in their song traits and the time and frequency of their chorus, specifically since birds use different sounds to communicate, mate, and defend breeding territories and rely on the vocal communication to sustain their lives [143, 214].

One of the main challenges in the majority of bioacoustics and avian behavior studies is the costly and time-consuming nature of working with audio data, which limits the duration and geographical extent of the research. The application of machine learning in bird song classification is not new [141], but most of the developed models are trained using specific sets of data, limiting the user to a pre-defined set of labels, with no control over what the model perceives as bird songs. This is specifically important in bird song studies since the model can classify some sounds, such as whistling, as bird sounds and discard some bird songs which are very different from what it was trained on [250]. In this case study, we demonstrate how Urban Rhapsody can facilitate such studies by providing a robust and easy to use solution where the user can search for specific sounds among hundreds of hours of recordings, refine the results if needed to reach the confidence level of interest, monitor the frequency and changes in the song traits, and investigate the impact of anthropogenic noises on birds. Sitting on the Atlantic Flyway, NYC offers great resting grounds for birds traveling along the north-south migratory route in the Americas [55]. We choose Washington Square Park, a popular local park situated in a dense and

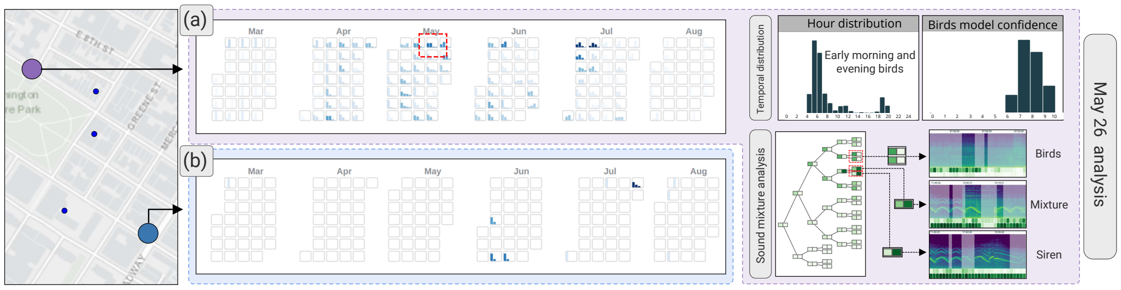


Figure 5.5: Looking for bird songs in two different Manhattan locations: (a) Edge of Washington Square Park with high concentration of bird songs and (b) a street corner on Broadway with very few instances of bird songs since we do not have trees for birds to nest.

busy neighborhood of the Manhattan borough, with the natural environment for birds to nest as well as the attributes representing a crowded and noisy urban environment [243]. The first step is to build our bird representation model. We start our exploration by using one of the bird song examples provided in the query view. Next, we select a day with high density of similar bird sounds. As shown in Figure 5.4(a), we generate a UMAP projection of our selected day on the Day View and see that majority of the bird songs are clustered on the bottom region of the projection (blue points). Next, we create our first representation model of bird songs to speed up our search across different days **(R1)**. We can find false positives and false negative examples throughout this process, fix those and refine our prototype. For instance, we found out that on April 18th, the model assigned a high likelihood to a small cluster of points (Figure 5.4(b)). We investigate this cluster closely and realize they are not bird songs, so we re-label these points, refine our model, make a new prediction with updated weights, and run this process iteratively until the model reaches a robust state **(R2)**. In the Model Summary View (Figure 5.4(c)), we can see that our new prototypes are converging: Our first model had the worst performance, and as we continued refining, the difference between the prediction probabilities of the labeled birds' data set get smaller after each iteration **(R3)**.

Using our refined model, we run a new query to explore the distribution and patterns of bird songs near Washington Square Park over the course of one year. The retrieved results clearly show two levels of seasonal patterns: a daily pattern with peaks in the mornings and afternoons corresponding to the dawn and dusk chorus times, and another pattern with peaks during spring to early summer, when songbirds usually migrate, as illustrated in Figure 5.5(a). This signifies the robust performance of the model in classifying birds. We also look at the corner of Broadway and Waverly Pl., where we have no trees on both sides of the street, to see if we can find similar patterns there. As Figure 5.5(b) shows, we have very few instances of bird songs in that location throughout 2017.

One useful aspect of Urban Rhapsody is the ability to analyze sound mixtures. To investigate how the siren sound can impact or even halt the birds' chorus, we use Urban Rhapsody to query for dawn chorus times (6-11 AM) where siren was also present. This allows us to discover whether loud sirens can halt birds' dawn chorus or whether birds in noisy urban areas like Manhattan local parks are adapted to the level of noise [173, 174]. We can use the Mixture Explorer to differentiate between these two sounds, as illustrated by Figure 5.5(a, bottom right). Notice that nodes containing bird songs, siren, or mixture of both are clearly distinguishable with our visual encodings **(R4)**. Drilling down to this specific example (Figure 5.6), we can see that the birds continue singing despite the loud siren **(R5)**. This analysis can create a ground for further research by bioacousticians and researchers in this field to investigate whether this pattern is more prevalent in birds of specific species or whether we can find incidents of ambient noise halting birds singing. Urban Rhapsody helped us to iteratively refine our model, track the sounds of interest and search for a combination of sounds across a large data set, detect the pattern

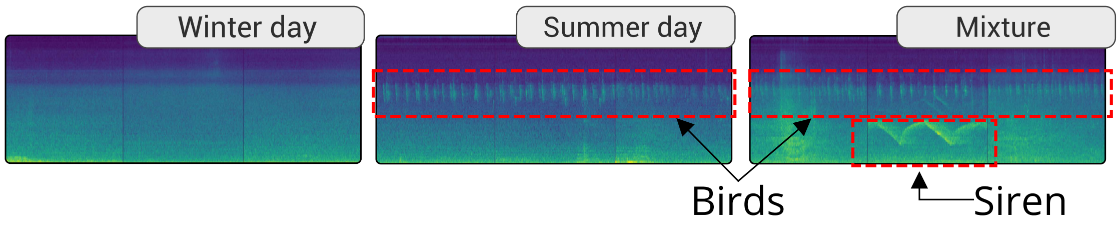


Figure 5.6: Spectrogram showing a winter day with no bird song, a summer day with birds' singing and the selected day in summer when birds dawn chorus continued despite loud siren.

and drill down to the exact moments to listen and investigate more.

5.8 Discussion and Conclusion

We have presented Urban Rhapsody, a novel interactive system for seamlessly exploring large audio data sets, based on user-generated concepts. Leveraging machine learning techniques, Urban Rhapsody supports labeling and analysis at scale, while our multilevel visualization approach enables the inspection of temporal patterns at varying levels of granularity. By enabling users to interactively label data based on their knowledge, Urban Rhapsody can be used to augment self-supervised methods that might not account for audio complexity. We illustrate its potential through data collected by the SONYC project. However, Urban Rhapsody can be applied to other longitudinal spatiotemporal acoustic data (e.g., bioacustics [68, 149]), and to support this we made the tool available on GitHub. We hope this will encourage researchers to use it in many different contexts and further develop the code base.

Limitations. While we define interactivity based on benchmarks for querying large data [14], we also identify three potential bottlenecks: similarity search, model training, and projection generation. Urban Rhapsody responds to similarity queries by returning up to 10,000 points in less than one second (for the examples provided as initial query seeds [67]). However, a one-time preprocessing computation is required to generate indices. This takes on average one hour per sensor/year and needs 9 GB of memory space (for sensors with low rates of missing data). GPU implementations [197, 198] achieve response times of under one second when loading Day View selections and for inference of created concept models. Deploying Urban Rhapsody to handle data from alternate sensor networks requires sufficient memory space to handle query indices, GPU capabilities to train models, and connectivity to support client-server architectures.

Expert feedback. Analyzing large collections of audio data is a challenging task, in which views into the data can be limited. The number of classes classifiers detect may be small, not matched to the task at hand, or too coarse-grained. Deep audio embeddings help to distill the semantics of audio to a smaller number of dimensions, but they are still very opaque and not easily interpretable. In addition, translation between modalities (e.g., using visual tools to explore audio data) is also highly challenging, and yet we know that it can be very effective. Our collaborators highlighted that Urban Rhapsody helps overcome these challenges by enabling interactive exploration, labeling, clustering, and reprojection of collections of audio data; and supports insights into models, labeled data, and previously unseen patterns within unlabeled data.

Future work. We plan to investigate whether Urban Rhapsody can accurately and efficiently represent concepts matching the user's mental model of their data. To investigate this we plan to conduct a large-scale user study with machine learning and audio researchers. While previous research [38] shows that spectrogram visualizations lead to high annotation accuracy at low time and labor costs, further investigation is also needed to explore additional visualization metaphors (e.g., to summarize longer periods of audio recordings). We will also explore how the analyses supported by systems such as Urban Rhapsody can useful to public officials and community representatives.

Conclusion. Urban Rhapsody is an interactive visual analytics tool for gaining insight into large collections of audio data, which we have demonstrated through use cases that characterize the acoustic environment of NYC. We believe that Urban Rhapsody offers an important step in moving beyond simple metrics, such as SPL, and will be of value to researchers in human-centered machine learning, acoustics, and urban science.

Chapter 6

Conclusions and Future Work

6.1 Summary of Work

This thesis proposes several interactive frameworks to tackle the growing challenge of interpreting the complex, heterogeneous data streams captured by ubiquitous sensors. The work puts forward a series of novel interactive frameworks designed to empower domain specialists to explore and derive insights from this data. The primary contributions include: ARGUS, a visual analytics system for debugging the intricate interplay between human actions and AI model outputs in augmented reality task-guidance systems; the StreetAware Dataset, a unique high-resolution, synchronized, multimodal collection of video, audio, and LiDAR data from New York City intersections designed to study complex urban dynamics; Crossroads, a pedestrian-focused visual analytics system that uses the StreetAware data to analyze intersection safety through automatic data enrichment and human-in-the-loop trajectory refinement; and Urban Rhapsody, a framework for the large-scale interactive exploration and classification of urban soundscapes. Together, these contributions advance the overarching goal of developing specialized visual analytics tools that make complex, real-world multimodal data accessible and actionable for experts across various domains.

6.2 Future work

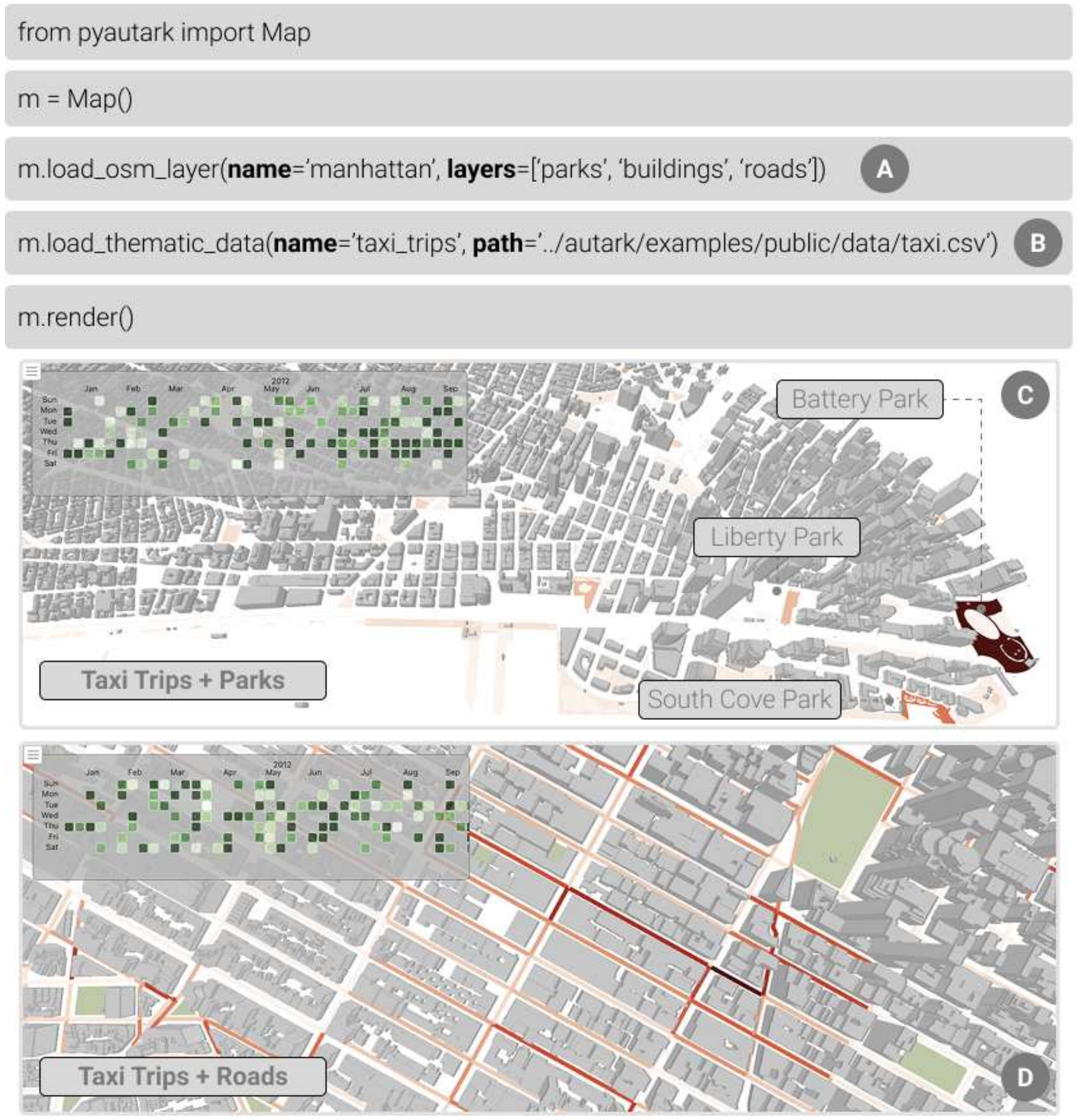
The work in this thesis aligns with a broader effort within the visualization community to develop powerful yet accessible visual analytics (VA) systems. The goal of this movement is to bridge the gap between complex data and human understanding by combining automated computational methods with interactive visual interfaces. However, the creation of such systems is a challenging task. It demands a wide range of skills, spanning from software engineering and algorithm design to user-centered design. Also, this process must be done in close partnership with domain specialists. The result of such a complex process is that the field is populated with many bespoke VA systems built for a single purpose. While often effective for their initial goal, these one-off solutions are frequently difficult to adapt, reproduce, or build upon, which ultimately reduces their long-term value and slows down cumulative progress in the field and its application to other disciplines.

Over the past two decades, the visualization community has made significant advancements in developing tools to facilitate the creation of custom data visualizations. This ecosystem ranges from low-level, expressive libraries like [26], which allow developers a more flexible control over SVG-based graphics, to high-level declarative grammars like Vega-Lite [**?**], which enable the rapid generation of complex charts from concise JSON specifications. Other notable toolkits such as Plotly, ECharts, and deck.gl have further democratized visualization development by providing pre-packaged components for common chart types and high-performance geospatial rendering. While these general-purpose tools are undeniably powerful and flexible, they are not inherently optimized for the unique demands of spatiotemporal data, a cornerstone of Urban Analytics. Analyzing this type of data effectively often requires highly interactive and coordinated multiple-view interfaces, where selections in one view (e.g., a specific time span on a timeline) dynamically filter and update the data displayed in another (e.g., a map showing geographic locations). Implementing this brushing-and-linking behavior across spatial and temporal dimensions requires significant engineering, presenting a steep learning curve and a substantial development burden that hinders rapid, exploratory analysis.

A further challenge lies in the disconnect between these visualization libraries and the standard workflow of a data scientist. Jupyter notebooks have become the de facto environment for data exploration, modeling, and analysis. While several libraries, such as anywidgets, ipywidgets, bqplot, and ipyleaflet, have emerged to enable interactive visualizations directly within a notebook, they often provide low-level components. A data scientist wishing to build querying capabilities on spatiotemporal data must still manually wire together multiple interactive widgets, managing state and event handling. This development effort detracts from the primary task of data analysis and creates a significant barrier. This challenge creates a need for a high-level paradigm that seamlessly integrates into the notebook environment, allowing data scientists to generate interactive spatiotemporal visualizations with minimal code and without having to switch contexts to a full-fledged web development environment.

6.2.1 PyAutark: A High-level Approach to Create Spatiotemporal Urban Visualizations

To start addressing the challenge of creating more reusable and accessible VA systems for spatiotemporal data, we have been designing a novel high-level Python library to bridge the gap between urban data analysis and interactive visualization. PyAutark provides a declarative paradigm tailored explicitly for creating visual analytics applications for spatiotemporal data directly within a data scientist's native environment. The library is built to empower urban data scientists to rapidly develop and deploy coordinated multi-view interfaces for exploring city-scale datasets, such as taxi trips, with just a few lines of Python code, lowering the barrier to entry for creating exploratory tools. This uses the urban domain as a primary testbed—a compelling choice for several reasons. From a societal perspective, with the majority of the world's population projected to live in cities by 2050, urban challenges like transportation and pollution are becoming increasingly critical. From a domain perspective, the interconnected nature of urban problems necessitates the integration of diverse datasets, making them ideal candidates for visual analytics. Indeed, VA systems have been instrumental in tackling a wide range of urban issues, yet this domain also exemplifies the problem of bespoke, hard-to-replicate solutions.



```
from pyautark import Map
m = Map()
m.load_osm_layer(name='manhattan', layers=['parks', 'buildings', 'roads'])
m.load_thematic_data(name='taxi_trips', path='../autark/examples/public/data/taxi.csv')
m.render()
```

Figure 6.1: Usage scenario of PyAutark. In a few lines of code, users can explore urban spatiotemporal datasets supported by city infrastructure data. A and B show the code needed to generate the interactive spatiotemporal visualization provided by PyAutark. C shows the distribution of taxi pickups around parks in lower Manhattan. D shows taxi trip data projected onto Manhattan's road network.

The main principle of PyAutark is its ability to contextualize thematic data by projecting it onto physical urban layers. The library's design is built around the concept of mapping geolocated datasets—such as taxi trips, noise complaints, or crime occurrences—onto spatial canvases that represent the city structure. These canvases can be 2D layers, like road networks or neighborhood polygons, or rich 3D layers representing building geometries. PyAutark's main capability is to abstract away the complex spatial computations (e.g., point-in-polygon aggregation, spatial joins) and rendering logic required for these projections. This allows a data scientist to, for example, take a raw dataset of taxi trips and, with a single command, visualize the trip density aggregated by neighborhood, or project pick-up and drop-off counts directly onto a 3D model of the city's buildings. This powerful feature of mapping dynamic data onto static physical layers provides immediate spatial context, transforming raw data points into an interpretable urban narrative and setting the stage for a more intuitive and powerful analytical workflow.

Figure 6.1 shows how PyAutark can be used to facilitate the exploration of spatiotemporal datasets. In this example, the user, working on a Jupyter Notebook environment, is interested in understanding the distribution of taxi pickups in Manhattan. The user starts by selecting the urban layers they are interested in using as a canvas to project their thematic data, taxi pickups in this example. Fig. 6.1-A shows that the user selected three different layers (parks, buildings, roads) that will be automatically downloaded from OpenStreetMaps using the Overpass API. Following the definition of the urban layers of interest, the user loads their thematic data by providing a name to the dataset and the location of the data on their local machine (Fig. 6.1-B). Once the user tells PyAutark to render the current configuration, an interactive visualization will be displayed in the notebook cell output. PyAutark will create a calendar widget in the top-right corner, allowing users to look at the temporal distribution of the taxi trips. Once a cell representing a day is clicked on the calendar widget, the taxi pickups will be projected onto one of the urban layers based on the user's selection. In (Fig. 6.1-C), the user's intention was to understand which park has the most number of taxi pickups around it, by projecting the taxi pickups data onto the closest park area for each trip, PyAutark colors the park areas with a color scale representing the density of taxi pickups. It becomes visible that Battery Park is the park that attracts the highest number of taxi pickups. The same can be done by projecting the taxi pickup data to the road network, as presented in Fig. 6.1-D.

The development of PyAutark represents a first step toward a general-purpose tool designed to democratize the exploration of spatiotemporal data. By providing a high-level paradigm that integrates directly into the data science ecosystem, it begins to lower the barriers that have historically made sophisticated visual analytics an exclusive domain of visualization experts. However, this is an ongoing effort. For PyAutark to fulfill its potential, future work will focus on expanding its core capabilities. Key priorities include adding support for flexible temporal aggregations—allowing users to seamlessly switch between daily, weekly, and monthly views—and enabling the composition of more complex dashboards with multiple linked widgets. As the library evolves, the ultimate goal is to foster a robust, community-driven tool that empowers researchers and practitioners across disciplines to unlock the insights hidden within their spatiotemporal data.

Bibliography

[1] Gstreamer. accessed on 20 February 2023.

[2] Technology for a quieter America, National Academy of Engineering. Technical report, Technical report, NAEPR-06-01-A, 2007.

[3] F. H. Administration. Navigating safety at unsignalized intersections. *Public Roads*, 79(4), January/February 2016.

[4] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking. *arXiv preprint arXiv:2206.14651*, 2022.

[5] S. Ahmed, M. Huda, S. Rajbhandari, C. Saha, M. Elshaw, and S. Kanarachos. Pedestrian and cyclist detection and intent estimation for autonomous vehicles: A survey. *Appl. Sci.*, 9(11):2335, 2019.

[6] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Human–Computer Interaction Series. Springer-Verlag, London, 2011.

[7] G. Andrienko and N. Andrienko. Spatio-temporal aggregation for visual analysis of movements. In *2008 IEEE Symposium on Visual Analytics Science and Technology*, pages 51–58. IEEE, 2008.

[8] R. Arandjelovic and A. Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617, 2017.

[9] J. Arellana, S. Fernández, M. Figueroa, and V. Cantillo. Analyzing pedestrian behavior when crossing urban roads by combining rp and sp

data. *Transportation research part F: traffic psychology and behaviour*, 85:259–275, 2022.

[10] ARGUS. Augmented reality guidance and user-modeling system. `https://github.com/VIDA-NYU/ARGUS`, 2023.

[11] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. *arXiv preprint ID:1610.09001*, 2016.

[12] A. L. Ballardini, A. Hernandez Saz, S. Carrasco Limeros, J. Lorenzo, I. Parra Alonso, N. Hernandez Parra, I. García Daza, and M. A. Sotelo. Urban intersection classification: A comparative analysis. *Sensors*, 21(18):6269, 2021.

[13] T. Banerjee, K. Chen, A. Almaraz, R. Sengupta, Y. Karnati, B. Grame, E. Posadas, S. Poddar, R. Schenck, J. Dilmore, S. Srinivasan, A. Rangarajan, and S. Ranka. A modern intersection data analytics system for pedestrian and vehicular safety. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3117–3124, 2022.

[14] L. Battle, P. Eichmann, M. Angelini, T. Catarci, G. Santucci, Y. Zheng, C. Binnig, J.-D. Fekete, and D. Moritz. Database benchmarking for supporting real-time interactive querying of large data. In *Proceedings of the 2020 International Conference on Management of Data*, SIGMOD '20, pages 1571–1587. ACM, 2020.

[15] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[16] R. Beams, E. Brown, W.-C. Cheng, J. S. Joyner, A. S. Kim, K. Kontson, D. Amiras, T. Baeuerle, W. Greenleaf, R. J. Grossmann, A. Gupta, C. Hamilton, H. Hua, T. T. Huynh, C. Leuze, S. B. Murthi, J. Penczek, J. Silva, B. Spiegel, A. Varshney, and A. Badano. Evaluation Challenges for the Application of Extended Reality Devices in Medicine. *Journal of Digital Imaging*, 35(5):1409–1418, 2022.

[17] M. Becher, D. Herr, C. Müller, K. Kurzhals, G. Reina, L. Wagner, T. Ertl, and D. Weiskopf. Situated Visual Analysis and Live Monitoring for Manufacturing. *IEEE Computer Graphics and Applications*, 42(2):33–44, 2022.

[18] J. P. Bello, C. Silva, O. Nov, R. L. Dubois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy. Sonyc: A system for monitoring, analyzing, and mitigating urban noise pollution. *Communications of the ACM*, 62(2):68–77, 2019.

[19] A. Ben Khalifa, I. Alouani, M. Mahjoub, and A. Rivenq. A novel multi-view pedestrian detection database for collaborative intelligent transportation systems. *Future Gener. Comput. Syst.*, 113:506–527, 2020.

[20] B. Benjdira, A. Koubaa, A. T. Azar, Z. Khan, A. Ammar, and W. Boulila. Tau: A framework for video-based traffic analytics leveraging artificial intelligence and unmanned aerial systems. *Engineering applications of artificial intelligence*, 114:105095, 2022.

[21] J. Bernard, M. Hutter, M. Zeppelzauer, D. Fellner, and M. Sedlmair. Comparing visual-interactive labeling with active learning: An experimental study. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):298–308, 2017.

[22] I. M. Bernhoft and G. Carstensen. Preferences and behaviour of pedestrians and cyclists by age and gender. *Transportation Research Part F: Traffic Psychology and Behaviour*, 11(2):83–95, 2008.

[23] H. Bi, T. Mao, Z. Wang, and Z. Deng. A deep learning-based framework for intersectional traffic simulation and editing. *IEEE Transactions on Visualization and Computer Graphics*, 26(7):2335–2348, 2020.

[24] D. Bohus, S. Andrist, A. Feniello, N. Saw, M. Jalobeanu, P. Sweeney, A. L. Thompson, and E. Horvitz. Platform for situated intelligence. *CoRR*, abs/2103.15975, 2021.

[25] M. Bostock. D3.js. `https://d3js.org/`.

[26] M. Bostock, V. Ogievetsky, and J. Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.

[27] E. Bozkir, O. Günlü, W. Fuhl, R. F. Schaefer, and E. Kasneci. Differential privacy for eye tracking with temporal correlations. *PLOS ONE*, 16(8):1–22, 2021.

[28] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila. The eurocity persons dataset: A novel benchmark for object detection. *arXiv*, 2018.

[29] A. Bronzaft. Neighborhood noise and its consequences. *Survey Research Unit, School of Public Affairs, Baruch College, New York*, 2007.

[30] A. L. Bronzaft and L. Hagler. Noise: The invisible pollutant that cannot be ignored. In *Emerging Environmental Technologies, Volume II*, pages 75–96. Springer, 2010.

[31] A. L. Brown. Soundscapes and environmental noise management. *Noise Control Engineering Journal*, 58(5):493–500, 2010.

[32] A. L. Brown. A review of progress in soundscapes and an approach to soundscape planning. *International Journal of Acoustics and Vibration*, 17(2):73–81, 2012.

[33] Q. Bui and E. Badger. The Coronavirus Quieted City Noise. Listen to What's Left. *The New York Times*, May 2020.

[34] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, Long Beach, CA, USA, June 2019.

[35] Z. Cai and N. Vasconcelos. Cascade R-CNN: delving into high quality object detection. *arXiv*, 2017.

[36] M. Cartwright, J. Cramer, A. E. M. Méndez, Y. Wang, H.-H. Wu, V. Lostanlen, M. Fuentes, G. Dove, C. Mydlarz, J. Salamon, et al. SONYC-UST-V2: an urban sound tagging dataset with spatiotemporal context. *arXiv*, 2020.

[37] M. Cartwright, J. Cramer, J. Salamon, and J. P. Bello. TriCycle: Audio representation learning from sensor network data using self-supervision. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 278–282. IEEE, 2019.

[38] M. Cartwright, A. Seals, J. Salamon, A. Williams, S. Mikloska, D. MacConnell, E. Law, J. P. Bello, and O. Nov. Seeing sound: Investigating the effects of visualizations and complexity on crowdsourced audio annotations. *Proceedings of the ACM on Human-Computer Interaction*, 1(CSCW):1–21, 2017.

[39] T. Caudell and D. Mizell. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, volume ii, pages 659–669, 1992.

[40] A. Chakraborty, V. Stamatescu, S. C. Wong, G. B. Wigley, and D. A. Kearney. A data set for evaluating the performance of multi-class multi-object video tracking. In *Proceedings of the Automatic Target Recognition XXVII*, volume 10202 of *Cergy-Pontoise, France*, pages 112–120, Anaheim, CA, USA, April 2017. SPIE.

[41] P. Charitidis, S. Moschos, A. Pipertzis, I. J. Theologou, M. Michailidis, S. Doropoulos, C. Diou, and S. Vologiannidis. Streetscouting: A deep learning platform for automatic detection and geotagging of urban features from street-level images. *Appl. Sci.*, 13(1):266, 2023.

[42] L. Chen, Y. Lu, Q. Sheng, Y. Ye, R. Wang, and Y. Liu. Estimating pedestrian volume using street view images: A large-scale validation test. *Comput. Environ. Urban Syst.*, 81:101481, 2020.

[43] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv*, 2018.

[44] R. Chen, J. Hu, M. W. Levin, and D. Rey. Stability-based analysis of autonomous intersection management with pedestrians. *Transportation research part C: emerging technologies*, 114:463–483, 2020.

[45] B. Cheng, B. Xiao, J. Wang, H. Shi, T. S. Huang, and L. Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation, 2019.

[46] F. Chirigati, H. Doraiswamy, T. Damoulas, and J. Freire. Data polygamy: the many-many relationships among urban spatio-temporal data sets. In *Procedings of the 2016 International Conference on Management of Data*, pages 1011–1025, 2016.

[47] City Report, Inc. New york rolling out noise law, listening tech for souped-up speedsters. accessed on 16 January 2023.

[48] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, Las Vegas, NV, USA, June 2016.

[49] K. Corona, K. Osterdahl, R. Collins, and A. Hoogs. MEVA: A large-scale multiview, multimodal video dataset for activity detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1060–1068, Snowmass Village, CO, USA, March 2020.

[50] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello. Look, listen, and learn more: Design choices for deep audio embeddings. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3852–3856. IEEE, 2019.

[51] K. Dahmane, N. Essoukri Ben Amara, P. Duthon, F. Bernardin, M. Colomb, and F. Chausse. The cerema pedestrian database: A specific database in adverse weather conditions to evaluate computer vision pedestrian detectors. In *Proceedings of the 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, pages 472–477, Hammamet, Tunisia, December 2016.

[52] DARPA. Perceptually-enabled task guidance (PTG). `https://www.darpa.mil/program/perceptually-enabled-task-guidance`.

[53] B. David-John, D. Hosfelt, K. Butler, and E. Jain. A privacy-preserving approach to streaming eye-tracking data. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2555–2565, 2021.

[54] W. J. Davies, M. D. Adams, N. S. Bruce, R. Cain, A. Carlyle, P. Cusack, D. A. Hall, K. I. Hume, A. Irwin, and P. Jennings. Perception of soundscapes: An interdisciplinary approach. *Applied acoustics*, 74(2):224–231, 2013.

[55] L. Day and D. Riepe. *Field Guide to the Neighborhood Birds of New York City*. JHU Press, 2015.

[56] K. M. de Paiva Vianna, M. R. A. Cardoso, and R. M. C. Rodrigues. Noise pollution and annoyance: An urban soundscapes study. *Noise & Health*, 17(76):125, 2015.

[57] T. Dema, M. Brereton, J. L. Cappadonna, P. Roe, A. Truskinger, and J. Zhang. Collaborative exploration and sensemaking of big environmental sound data. *Computer Supported Cooperative Work*, 26(4–6):693–731, 2017.

[58] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou. Retinaface: Single-stage dense face localisation in the wild. *arXiv*, 2019.

[59] Z. Deng, D. Weng, S. Liu, Y. Tian, M. Xu, and Y. Wu. A survey of urban visual analytics: Advances and future directions. *Computational Visual Media*, 9(1):3–39, 2023.

[60] Z. Deng, D. Weng, X. Xie, J. Bao, Y. Zheng, M. Xu, W. Chen, and Y. Wu. Compass: Towards better causal analysis of urban time series. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1051–1061, 2021.

[61] D. Doiron, E. Setton, J. Brook, Y. Kestens, G. Mccormack, M. Winters, M. Shooshtari, S. Azami, and D. Fuller. Predicting walking-to-work using street-level imagery and deep learning in seven canadian cities. *Sci. Rep.*, 12:18380, 2022.

[62] H. Doraiswamy, J. Freire, M. Lage, F. Miranda, and C. Silva. Spatio-temporal urban data analysis: A visual analytics perspective. *IEEE Computer Graphics and Applications*, 38(5):26–35, 2018.

[63] H. Doraiswamy, E. Tzirita Zacharatou, F. Miranda, M. Lage, A. Ailamaki, C. T. Silva, and J. Freire. Interactive visual exploration of spatio-temporal urban data sets using urbane. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 1693–1696. ACM, 2018.

[64] J. Dratva, E. Zemp, D. F. Dietrich, P.-O. Bridevaux, T. Rochat, C. Schindler, and M. W. Gerbase. Impact of road traffic noise annoyance on health-related quality of life: Results from a population-based study. *Quality of Life Research*, 19(1):37–46, 2010.

[65] Y. Duan, C. Yang, H. Chen, W. Yan, and H. Li. Low-complexity point cloud denoising for lidar by pca-based dimension reduction. *Optics Communications*, 482:126567, 2021.

[66] S. El Hamdani, N. Benamar, and M. Younis. Pedestrian support in intelligent transportation systems: Challenges, solutions and open issues. *Transportation Research Part C: Emerging Technologies*, 121:102856, 2020.

[67] Faiss. URL: `https://faiss.ai/`.

[68] A. Farnsworth, S. Kelling, V. Lostanlen, J. Salamon, A. Cramer, and J. P. Bello. BirdVox-296h: a large-scale dataset for detection and classification of flight calls. Dec. 2021.

[69] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019.

[70] C. Felix, S. L. Franconeri, and E. Bertini. Taking word clouds apart: An empirical investigation of the design space for keyword summaries. *IEEE Transactions on Visualization and Computer Graphics*, 24:657–666, 2018.

[71] Z. Feng, H. Qu, S.-H. Yang, Y. Ding, and J. Song. A survey of visual analytics in urban area. *Expert Systems*, 39(9):e13065, 2022.

[72] I. Fernández del Amo, J. A. Erkoyuncu, R. Roy, and S. Wilding. Augmented Reality in Maintenance: An information-centred design framework. *Procedia Manufacturing*, 19:148–155, 2018.

[73] L. Ferreira, G. Moreira, M. Hosseini, M. Lage, N. Ferreira, and F. Miranda. Assessing the landscape of toolkits, frameworks, and authoring tools for urban visual analytics systems. *Computers & Graphics*, 123:104013, 2024.

[74] N. Ferreira, M. Lage, H. Doraiswamy, H. Vo, L. Wilson, H. Werner, M. Park, and C. Silva. Urbane: A 3D framework to support data driven decision making in urban development. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 97–104. IEEE, 2015.

[75] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva. Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013.

[76] P. Fleck, A. Sousa Calepso, S. Hubenschmid, M. Sedlmair, and D. Schmalstieg. RagRug: A Toolkit for Situated Analytics. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2022.

[77] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra. FSD50k: an open dataset of human-labeled sound events. *arXiv preprint ID:2010.00475*, 2020.

[78] M. Fourkiotis, C. Kazaklari, A. Kopsacheilis, and I. Politis. Applying deep learning techniques for the prediction of pedestrian behaviour on crossings with countdown signal timers. *Transp. Res. Procedia*, 60:536–543, 2022.

[79] Foxglove. Foxglove - Visualizing and debugging your robotics data.

[80] M. Fuentes, B. Steers, P. Zinemanas, M. Rocamora, L. Bondi, J. Wilkins, Q. Shi, Y. Hou, S. Das, X. Serra, et al. Urban sound & sight: Dataset and benchmark for audio–visual urban scene understanding. In *Proceedings of the ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 141–145, Singapore, May 2022.

[81] M. Funk, S. Mayer, and A. Schmidt. Using in-situ projection to support cognitively impaired workers at the workplace. In *Proceedings of the 17th international ACM SIGACCESS conference on Computers & accessibility*, pages 185–192, 2015.

[82] A. Fyhri, H. B. Sundfør, T. Bjørnskau, and A. Laureshyn. Safety in numbers for cyclists—conclusions from a multidisciplinary study of seasonal change in interplay and conflicts. *Accident Analysis & Prevention*, 105:124–133, 2017.

[83] T. Gandhi and M. M. Trivedi. Pedestrian protection systems: Issues, survey, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 8(3):413–430, 2007.

[84] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, Providence, RI, USA, June 2012.

[85] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE, 2017.

[86] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780. IEEE, 2017.

[87] R. Girdhar, M. Singh, N. Ravi, L. van der Maaten, A. Joulin, and I. Misra. Omnivore: A single model for many visual modalities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16102–16112, 2022.

[88] R. Girshick. Fast R-CNN. *arXiv*, 2015.

[89] Google LLC. Google street view. accessed on 20 February 2023.

[90] L. Gou, L. Zou, N. Li, M. Hofmann, A. K. Shekar, A. Wendt, and L. Ren. VATLD: A visual analytics system to assess, understand and improve traffic light detection. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):261–271, 2021.

[91] S. Grollmisch, E. Cano, C. Kehling, and M. Taenzer. Analyzing the Potential of Pre-Trained Embeddings for Audio Classification Tasks. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 790–794. IEEE, 2021.

[92] C. Guastavino. *Etude sémantique et acoustique de la perception des basses fréquences dans l'environnement sonore urbain.* PhD Thesis, Paris 6, 2003.

[93] H. F. Guite, C. Clark, and G. Ackrill. The impact of the physical and urban environment on mental well-being. *Public Health*, 120(12):1117–1126, 2006.

[94] H. Guo, Z. Wang, B. Yu, H. Zhao, and X. Yuan. Tripvista: Triple perspective visual trajectory analytics and its application on microscopic traffic data at a road intersection. In *2011 IEEE Pacific Visualization Symposium*, pages 163–170, 2011.

[95] M. S. Hammer, T. K. Swinburn, and R. L. Neitzel. Environmental noise pollution in the United States: developing an effective public health response. *Environmental Health Perspectives*, 122(2):115–119, 2014.

[96] A. S. Haralabidis, K. Dimakopoulou, F. Vigna-Taglianti, M. Giampaolo, A. Borgini, M.-L. Dudley, G. Pershagen, G. Bluhm, D. Houthuijs, and W. Babisch. Acute effects of night-time noise exposure on blood pressure in populations living near airports. *European Heart Journal*, 29(5):658–664, 2008.

[97] J. M. Hausdorff, D. A. Rios, and H. K. Edelberg. Gait variability and fall risk in community-living older adults: A 1-year prospective study. *Archives of Physical Medicine and Rehabilitation*, 82(8):1050–1056, 2001.

[98] N. L. Haworth and A. Schramm. Illegal and risky riding of electric scooters in brisbane. *Medical Journal of Australia*, 211(9):412–413, 2019.

[99] W. He, L. Zou, A. K. Shekar, L. Gou, and L. Ren. Where can we help? a visual analytics approach to diagnosing and improving semantic segmentation of movable objects. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1040–1050, 2022.

[100] S. Henderson and S. Feiner. Exploring the Benefits of Augmented Reality Documentation for Maintenance and Repair. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1355–1368, 2011.

[101] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, and others. CNN architectures for large-scale audio classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135. IEEE, 2017.

[102] M. M. Howlader, Y. Ali, A. Burbridge, and M. M. Haque. Before-after safety evaluation of part-time protected right-turn signals: An extreme value theory approach by applying artificial intelligence-based video analytics. *Accident Analysis & Prevention*, 194:107341, 2024.

[103] C.-T. Hsu and Y.-C. Tsan. Mosaics of video sequences with moving objects. *Signal Processing: Image Communication*, 19(1):81–98, 2004.

[104] Insurance Institute for Highway Safety. Pedestrian fatality statistics, 2023. Accessed: 17 March 2024.

[105] M. Itoh, D. Yokoyama, M. Toyoda, Y. Tomita, S. Kawamura, and M. Kitsuregawa. Visual fusion of mega-city big data: an application to traffic and tweets data analysis of metro passengers. In *2014 IEEE International Conference on Big Data (Big Data)*, pages 431–440. IEEE, 2014.

[106] S. Jamonnak, Y. Zhao, X. Huang, and M. Amiruzzaman. Geo-context aware study of vision-based autonomous driving models and spatial video data. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1019–1029, 2022.

[107] A. Jansen, M. Plakal, R. Pandya, D. P. Ellis, S. Hershey, J. Liu, R. C. Moore, and R. A. Saurous. Unsupervised learning of semantic audio representations. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 126–130. IEEE, 2018.

[108] T. Jiang, D. Yu, Y. Wang, T. Zan, S. Wang, and Q. Li. HoloLens-Based Vascular Localization System. *Journal of Medical Internet Research*, 22(4):e16852, Apr. 2020.

[109] G. Jocher, A. Chaurasia, and J. Qiu. Ultralytics YOLO. Jan. 2023.

[110] K. Johnston, J. M. Ver Hoef, K. Krivoruchko, and N. Lucas. *Using ArcGIS geostatistical analyst*, volume 380. Esri Redlands, 2001.

[111] P. Joia, D. Coimbra, J. A. Cuminato, F. V. Paulovich, and L. G. Nonato. Local affine multidimensional projection. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2563–2571, 2011.

[112] A. Kathuria and P. Vedagiri. Evaluating pedestrian vehicle interaction dynamics at un-signalized intersections: A proactive approach for safety analysis. *Accident Analysis & Prevention*, 134:105316, 2020.

[113] E. Kazakos, A. Nagrani, A. Zisserman, and D. Damen. Slow-fast auditory streams for audio recognition. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 855–859. IEEE, 2021.

[114] J. Kehrer and H. Hauser. Visualization and Visual Analysis of Multifaceted Scientific Data: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):495–513, 2013.

[115] O. Kilani, M. Gouda, J. Weiß, and K. El-Basyouny. Safety assessment of urban intersection sight distance using mobile lidar data. *Sustainability*, 13(16):9259, 2021.

[116] K. Kim, L. Boelling, S. Haesler, J. Bailenson, G. Bruder, and G. F. Welch. Does a Digital Assistant Need a Body? The Influence of Visual Embodiment and Social Behavior on the Perception of Intelligent Virtual Agents in AR. In *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 105–114, 2018. ISSN: 1554-7868.

[117] R. Korbmacher and A. Tordeux. Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches. *IEEE Trans. Intell. Transp. Syst.*, 23:24126–24144, 2022.

[118] Z. Kostić, A. Angus, Z. Yang, Z. Duan, I. Seskar, G. Zussman, and D. Raychaudhuri. Smart city intersections: Intelligence nodes for future metropolises. *Computer*, 55:74–85, 2022.

[119] S. Kreiss, L. Bertoni, and A. Alahi. OpenPifPaf: Composite Fields for Semantic Keypoint Detection and Spatio-Temporal Association. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–14, March 2021.

[120] A. Kumar, M. Khadkevich, and C. Fügen. Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 326–330. IEEE, 2018.

[121] C. Lee, Y. Kim, S. Jin, D. Kim, R. Maciejewski, D. Ebert, and S. Ko. A visual analytics system for exploring, monitoring, and forecasting road traffic congestion. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3133–3146, 2020.

[122] M. Lenormand, B. Gonçalves, A. Tugores, and J. J. Ramasco. Human diffusion and city influence. *Journal of The Royal Society Interface*, 12(109):20150473, 2015.

[123] J. Li, L. Liu, H. Xu, S. Wu, and C. J. Xue. Cross-camera inference on the constrained edge. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2023.

[124] Z. Liao, Y. Yu, and B. Chen. Anomaly detection in gps data based on visual analytics. In *2010 IEEE Symposium on Visual Analytics Science and Technology*, pages 51–58. IEEE, 2010.

[125] A. Lin, S. Rao, A. Celikyilmaz, E. Nouri, C. Brockett, D. Dey, and W. B. Dolan. A recipe for creating multimodal aligned datasets for sequential tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4871–4884, 2020.

[126] H. Lin, S. Gao, D. Gotz, F. Du, J. He, and N. Cao. Rclens: Interactive rare category exploration and identification. *IEEE Transactions on Visualization and Computer Graphics*, 24(7):2223–2237, 2017.

[127] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *arXiv*, 2014.

[128] C.-F. Liu and P.-Y. Chiang. Smart glasses based intelligent trainer for factory new recruits. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, MobileHCI '18, pages 395–399. Association for Computing Machinery, 2018.

[129] S. Liu, D. Maljovec, B. Wang, P.-T. Bremer, and V. Pascucci. Visualizing High-Dimensional Data: Advances in the Past Decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268, 2017.

[130] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. Berg. SSD: single shot multibox detector. *arXiv*, 2015.

[131] Y. Liu, E. Jun, Q. Li, and J. Heer. Latent space cartography: Visual analysis of vector space embeddings. *Computer Graphics Forum*, 38(3):67–78, 2019.

[132] Z. Liu and J. Heer. The effects of interactive latency on exploratory visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2122–2131, 2014.

[133] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*. IEEE, 1999.

[134] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[135] S. Lumnitz, T. Devisscher, J. Mayaud, V. Radic, N. Coops, and V. Griess. Mapping trees along urban street networks with deep learning and street-level imagery. *ISPRS J. Photogramm. Remote. Sens.*, 175:144–157, 2021.

[136] W. Ma, L. Zou, K. An, N. H. Gartner, and M. Wang. A partition-enabled multi-mode band approach to arterial traffic signal optimization. *IEEE Transactions on Intelligent Transportation Systems*, 20(1):313–322, 2018.

[137] T. K. O. Madsen and H. Lahrmann. Comparison of five bicycle facility designs in signalized intersections using traffic conflict studies. *Transportation research part F: traffic psychology and behaviour*, 46:438–450, 2017.

[138] A. Malik, R. Maciejewski, N. Elmqvist, Y. Jang, D. S. Ebert, and W. Huang. A correlative analysis process in a visual analytics environment. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 33–42. IEEE, 2012.

[139] K. Marriott, F. Schreiber, T. Dwyer, K. Klein, N. H. Riche, T. Itoh, W. Stuerzlinger, and B. H. Thomas, editors. *Immersive Analytics*. Springer International Publishing, 2018.

[140] J. C. Mays. Why Construction Noise Is Keeping You Up at 3 A.M. *The New York Times*, Sept. 2019.

[141] A. McIlraith and H. Card. Bird song identification using artificial neural networks and statistical analysis. In *CCECE'97. Canadian Conference on Electrical and Computer Engineering. Engineering Innovation: Voyage of Discovery. Conference Proceedings*, volume 1, pages 63–66. IEEE, 1997.

[142] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint ID:1802.03426*, 2018.

[143] S. Mendes, V. J. Colino-Rabanal, and S. J. Peris. Bird song variations along an urban gradient: The case of the european blackbird (turdus merula). *Landscape and Urban Planning*, 99(1):51–57, 2011.

[144] J. Mesimäki and J. Luoma. Near accidents and collisions between pedestrians and cyclists. *European transport research review*, 13(1):38, 2021.

[145] Microsoft. Typescript. https://www.typescriptlang.org/.

[146] Microsoft. Using the windows device portal. https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/using-the-windows-device-portal, 2022.

[147] M. Miknis, R. Davies, P. Plassmann, and A. Ware. Near real-time point cloud processing using the pcl. In *2015 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 153–156. IEEE, 2015.

[148] P. Milgram and F. Kishino. A Taxonomy of Mixed Reality Visual Displays. *IEICE Transactions on Information Systems*, E77-D, 1994.

[149] B. S. Miller, M. Milnes, and S. Whiteside. Long-term underwater acoustic recordings 2013-2019. URL: https://researchdata.edu.au/long-term-underwater-2013-2019/967510.

[150] F. Miranda, H. Doraiswamy, M. Lage, L. Wilson, M. Hsieh, and C. T. Silva. Shadow Accrual Maps: Efficient accumulation of city-scale shadows over time. *IEEE Transactions on Visualization and Computer Graphics*, 25(3):1559–1574, 2019.

[151] F. Miranda, H. Doraiswamy, M. Lage, K. Zhao, B. Gonçalves, L. Wilson, M. Hsieh, and C. T. Silva. Urban Pulse: Capturing the rhythm of cities. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):791–800, 2017.

[152] F. Miranda, H. Doraiswamy, M. Lage, K. Zhao, B. Gonçalves, L. Wilson, M. Hsieh, and C. T. Silva. Urban Pulse: Capturing the rhythm of cities. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):791–800, 2017.

[153] F. Miranda, M. Hosseini, M. Lage, H. Doraiswamy, G. Dove, and C. T. Silva. Urban mosaic: Visual exploration of streetscapes using large-scale image data. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI'20, pages 1–15, New York, NY, USA, April 2020. Association for Computing Machinery.

[154] F. Miranda, M. Hosseini, M. Lage, H. Doraiswamy, G. Dove, and C. T. Silva. Urban mosaic: Visual exploration of streetscapes using large-scale image data. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–15, New York, NY, USA, 2020. Association for Computing Machinery.

[155] F. Miranda, M. Hosseini, M. Lage, H. Doraiswamy, G. Dove, and C. T. Silva. Urban Mosaic: Visual exploration of streetscapes using large-scale image data. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–15. ACM, 2020.

[156] F. Miranda, M. Lage, H. Doraiswamy, C. Mydlarz, J. Salamon, Y. Lockerman, J. Freire, and C. T. Silva. Time Lattice: A data structure for the interactive visual analysis of large time series. *Computer Graphics Forum*, 37(3):23–35, 2018.

[157] MIT Lincoln Laboratory. PTG evaluation tasks vol. 1. `TBD`, 2022.

[158] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications*, 2009.

[159] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.

[160] M. Muja and D. G. Lowe. Fast matching of binary features. In *Computer and Robot Vision (CRV)*, pages 404–410, 2012.

[161] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36, 2014.

[162] A. Muzet. The need for a specific noise measurement for population exposed to aircraft noise during night-time. *Noise and Health*, 4(15):61, 2002.

[163] M. S. Nabavi Niaki, N. Saunier, and L. F. Miranda-Moreno. Is that move safe? case study of cyclist movements at intersections with cycling discontinuities. *Accident Analysis & Prevention*, 131:239–247, 2019.

[164] M. Nadj, M. Knaeble, M. X. Li, and A. Maedche. Power to the oracle? Design principles for interactive labeling systems in machine learning. *KI-Künstliche Intelligenz*, 34(2):131–142, 2020.

[165] V. Nair, L. Rosenberg, J. F. O'Brien, and D. Song. Truth in motion: The unprecedented risks and opportunities of extended reality motion data. `https://arxiv.org/abs/2306.06459`, 2023.

[166] K. Namitha, A. Narayanan, and M. Geetha. Interactive visualization-based surveillance video synopsis. *Applied Intelligence*, 52(4):3954–3975, July 2021.

[167] A. Nandy, S. Chakraborty, J. Chakraborty, and G. Venture. *Modern Methods for Affordable Clinical Gait Analysis*. Academic Press, 2021.

[168] J. Nasar, P. Hecht, and R. Wener. Mobile telephones, distracted attention, and pedestrian safety. *Accident Analysis & Prevention*, 40(1):69–75, 2008.

[169] J. L. Nasar and D. Troyer. Pedestrian injuries due to mobile phone use in public places. *Accident Analysis & Prevention*, 57:91–95, 2013.

[170] A. Nassar. *Learning to Map Street-Side Objects Using Multiple Views*. PhD thesis, Université de Bretagne Sud, Brittany, France, 2021.

[171] B. Navarro, L. Miranda-Moreno, N. Saunier, A. Labbe, and T. Fu. Do stop-signs improve the safety for all road users? a before-after study of stop-controlled intersections using video-based trajectories and surrogate measures of safety. *Accident Analysis & Prevention*, 167:106563, 2022.

[172] R. L. Neitzel, R. R. Gershon, T. P. McAlexander, L. A. Magda, and J. M. Pearson. Exposures to transit and other sources of noise among New York City residents. *Environmental science & technology*, 46(1):500–508, 2012.

[173] E. Nemeth and H. Brumm. Birds and anthropogenic noise: are urban songs adaptive? *The American Naturalist*, 176(4):465–475, 2010.

[174] E. Nemeth, N. Pieretti, S. A. Zollinger, N. Geberzahn, J. Partecke, A. C. Miranda, and H. Brumm. Bird song and anthropogenic noise: vocal constraints may explain why birds sing higher-frequency songs in cities. *Proceedings of the Royal Society B: Biological Sciences*, 280(1754):20122798, 2013.

[175] L. Neumann, M. Karg, S. Zhang, C. Scharfenberger, E. Piegert, S. Mistr, O. Prokofyeva, R. Thiel, A. Vedaldi, A. Zisserman, et al. Nightowls: A pedestrians at night dataset. In *Proceedings of the Asian Conference on Computer Vision*, Berlin/Heidelberg, Germany, pages 691–705, Perth, WA, Australia, December 2018. Springer.

[176] Y. Ni, M. Wang, J. Sun, and K. Li. Evaluation of pedestrian safety at intersections: A theoretical framework based on pedestrian-vehicle interaction patterns. *Accident Analysis & Prevention*, 96:118–129, 2016.

[177] A. Nijholt. Towards Social Companions in Augmented Reality: Vision and Challenges. In N. A. Streitz and S. Konomi, editors, *Distributed, Ambient and Pervasive Interactions. Smart Living, Learning, Well-being and Health, Art and Creativity*, Lecture Notes in Computer Science, pages 304–319. Springer International Publishing, 2022.

[178] A. Noulas, S. Scellato, R. Lambiotte, M. Pontil, and C. Mascolo. A tale of many cities: universal patterns in human urban mobility. *PloS one*, 7(5):e37027, 2012.

[179] NVIDIA. Deepstream sdk. accessed on 31 January 2023.

[180] W. H. Organization. *Burden of disease from environmental noise: Quantification of healthy life years lost in Europe.* World Health Organization. Regional Office for Europe, 2011.

[181] W. H. Organization. *Helmets: a road safety manual for decision-makers and practitioners.* World Health Organization, 2nd ed edition, 2023.

[182] T. Ortner, J. Sorger, H. Steinlechner, G. Hesina, H. Piringer, and E. Gröller. Vis-a-ware: Integrating spatial and non-spatial visualization for visibility-aware urban planning. *IEEE Transactions on Visualization and Computer Graphics*, 23(2):1139–1151, 2016.

[183] S. Pase. Ethical considerations in augmented reality applications. In *Proceedings of the International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government (EEE)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2012.

[184] S. R. Payne, W. J. Davies, and M. D. Adams. Research into the practical and policy applications of soundscape concepts and techniques in urban areas. Technical report, University of Salford, 2009.

[185] Y. Piadyk, J. Rulff, E. Brewer, M. Hosseini, K. Ozbay, M. Sankaradas, S. Chakradhar, and C. Silva. Streetaware: A high-resolution synchronized multimodal urban scene dataset. *Sensors*, 23(7), 2023.

[186] Y. Piadyk, B. Steers, C. Mydlarz, M. Salman, M. Fuentes, J. Khan, H. Jiang, K. Ozbay, J. P. Bello, and C. Silva. Reip: A reconfigurable environmental intelligence platform and software framework for fast sensor network prototyping. *Sensors*, 22(10):3809, 2022.

[187] H. Piringer, M. Buchetics, and R. Benedik. AlVis: Situation awareness in the surveillance of road tunnels. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 153–162, 2012.

[188] B. Puladi, M. Ooms, M. Bellgardt, M. Cesov, M. Lipprandt, S. Raith, F. Peters, S. C. Möhlhenrich, A. Prescher, F. Hölzle, T. W. Kuhlen, and A. Modabber. Augmented Reality-Based Surgery on the Human Cadaver Using a New Generation of Optical Head-Mounted Displays. *JMIR Serious Games*, 10(2):e34781, 2022.

[189] K. Qinghong Lin, A. Jinpeng Wang, M. Soldan, M. Wray, R. Yan, E. Zhongcong Xu, D. Gao, R. Tu, W. Zhao, W. Kong, et al. Egocentric video-language pretraining. *arXiv e-prints*, pages arXiv–2206, 2022.

[190] D. Quercia and D. Saez. Mining urban deprivation from foursquare: Implicit crowdsourcing of city land use. *IEEE Pervasive Computing*, 13(2):30–36, 2014.

[191] R. Quintero Mínguez, D. Fernández-Llorca, and M.-A. Sotelo. Pedestrian intention and pose prediction through dynamical models and behaviour classification. 09 2015.

[192] R. Quintero Mínguez, I. Parra, and M.-A. Sotelo. Pedestrian path prediction based on body language and action classification. 10 2014.

[193] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[194] M. Raimbault and D. Dubois. Urban soundscapes: Experiences and knowledge. *Cities*, 22(5):339–350, 2005.

[195] M. Raimbault, C. Lavandier, and M. Bérengier. Ambient sound assessment of urban environments: field studies in two French cities. *Applied Acoustics*, 64(12):1241–1256, 2003.

[196] S. Ramírez. Fastapi. `https://fastapi.tiangolo.com/`.

[197] RAPIDS. URL: `https://rapids.ai/start.html`.

[198] RAPIDS Benchmark. URL: `https://www.alcf.anl.gov/sites/default/files/2021-03/NVIDIA_RAPIDS_ANL.pdf`.

[199] A. Rasouli, I. Kotseruba, T. Kunic, and J. Tsotsos. Pie: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6261–6270, Seoul, South Korea, October–November 2019.

[200] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[201] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.*, 28:1497, 2015.

[202] J. Rulff, F. Miranda, M. Hosseini, M. Lage, M. Cartwright, G. Dove, J. Bello, and C. T. Silva. Urban rhapsody: Large-scale exploration of urban soundscapes. *Computer Graphics Forum*, 41(3):209–221, 2022.

[203] A. Sainju and Z. Jiang. Mapping road safety features from streetview imagery: A deep learning approach. *ACM/IMS Trans. Data Sci.*, 1:1–20, 2020.

[204] A. Schmeil and W. Broll. MARA - A Mobile Augmented Reality-Based Virtual Assistant. In *2007 IEEE Virtual Reality Conference*, pages 267–270, 2007.

[205] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[206] G. Seress and A. Liker. Habitat urbanization and its effects on birds. *Acta Zoologica Academiae Scientiarum Hungaricae*, 61(4):373–408, 2015.

[207] M. S. Shirazi and B. Morris. Observing behaviors at intersections: A review of recent studies & developments. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1258–1263, 2015.

[208] M. S. Shirazi and B. T. Morris. Looking at intersections: A survey of intersection monitoring, behavior and safety analysis of recent studies. *IEEE Transactions on Intelligent Transportation Systems*, 18(1):4–24, 2017.

[209] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE symposium on visual languages*, pages 336–343. IEEE, 1996.

[210] R. Sicat, J. Li, J. Choi, M. Cordeil, W.-K. Jeong, B. Bach, and H. Pfister. DXR: A Toolkit for Building Immersive Data Visualizations. *IEEE*

Transactions on Visualization and Computer Graphics, 25(1):715–725, 2019. Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[211] B.-I. Sighencea, R.-I. Stanciu, and C.-D. Căleanu. A review of deep learning-based methods for pedestrian trajectory prediction. *Sensors*, 21(22):7543, 2021.

[212] N. Sikka, C. Vila, M. Stratton, M. Ghassemi, and A. Pourmand. Sharing the sidewalk: A case of e-scooter related pedestrian injury. *The American Journal of Emergency Medicine*, 37(9):1807.e5–1807.e7, 2019.

[213] K. K. Singh, K. Fatahalian, and A. A. Efros. Krishnacam: Using a longitudinal, single-person, egocentric dataset for scene understanding tasks. In *Proceedings of the 2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9, Lake Placid, NY, USA, March 2016.

[214] H. Slabbekoorn. Songs of the city: noise-dependent spectral plasticity in the acoustic phenotype of urban birds. *Animal Behaviour*, 85(5):1089–1099, 2013.

[215] D. Smilkov, N. Thorat, C. Nicholson, E. Reif, F. B. Viégas, and M. Wattenberg. Embedding projector: Interactive visualization and interpretation of embeddings. *arXiv preprint arXiv:1611.05469*, 2016.

[216] M. O. Source. React. `https://react.dev/`.

[217] D. Steedly, C. Pal, and R. Szeliski. Efficiently registering video into panoramic mosaics. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1300–1307 Vol. 2, 2005.

[218] J. Stipancic, L. Miranda-Moreno, J. Strauss, and A. Labbe. Pedestrian safety at signalized intersections: Modelling spatial effects of exposure, geometry and signalization on a large urban network. *Accident Analysis & Prevention*, 134:105265, 2020.

[219] M. Sukel, S. Rudinac, and M. Worring. Urban object detection kit: A system for collection and analysis of street-level imagery. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, ICMR'20, pages 509–516, New York, NY, USA, June 2020. Association for Computing Machinery.

[220] G. Sun, Y. Zhao, D. Cao, J. Li, R. Liang, and Y. Liu. AtoMixer: Atom-based interactive visual exploration of traffic surveillance data. *Journal of Computer Languages*, 53:53–62, 2019.

[221] K. Sun, B. Xiao, D. Liu, and J. Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.

[222] X. Sun, S. B. Murthi, G. Schwartzbauer, and A. Varshney. High-Precision 5DoF Tracking and Visualization of Catheter Placement in EVD of the Brain Using AR. *ACM Transactions on Computing for Healthcare*, 1(2):9:1–9:18, 2020.

[223] Sun, P., H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, Long Beach, CA, USA, June 2019.

[224] B. Szubert, J. E. Cole, C. Monaco, and I. Drozdov. Structure-preserving visualisation of high dimensional single-cell datasets. *Scientific reports*, 9(1):1–10, 2019.

[225] M. Tagliasacchi, B. Gfeller, F. de Chaumont Quitry, and D. Roblek. Pre-training audio representations with self-supervision. *IEEE Signal Processing Letters*, 27:600–604, 2020.

[226] A. Tang, C. Owen, F. Biocca, and W. Mou. Comparative effectiveness of augmented reality in object assembly. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 73–80, 2003.

[227] L. Tang, Y. Liu, J. Li, R. Qi, S. Zheng, B. Chen, and H. Yang. Pedestrian crossing design and analysis for symmetric intersections: Efficiency and safety. *Transportation research part A: policy and practice*, 142:187–206, 2020.

[228] M. S. Tarawneh. Evaluation of pedestrian speed in Jordan with investigation of some contributing factors. *Journal of Safety Research*, 32(2):229–236, 2001.

[229] three.js. three.js. `https://threejs.org/`.

[230] E. Tokuda, Y. Lockerman, G. Ferreira, E. Sorrelgreen, D. Boyle, R. Jr., and C. Silva. A new approach for pedestrian density estimation using moving sensors and computer vision. *arXiv*, 2018.

[231] A. Tordeux, M. Chraibi, A. Seyfried, and A. Schadschneider. Prediction of pedestrian speed with artificial neural networks. *arXiv*, 2018.

[232] D. Ungureanu, F. Bogo, S. Galliani, P. Sama, X. Duan, C. Meekhof, J. Stühmer, T. J. Cashman, B. Tekin, J. L. Schönberger, P. Olszta, and M. Pollefeys. Hololens 2 research mode as a tool for computer vision research. *CoRR*, abs/2008.11239, 2020.

[233] G. Valdrighi, N. Ferreira, and J. Poco. Morevis: A visual summary for spatiotemporal moving regions, 2023.

[234] E. Van Kempen, J. Devilee, W. Swart, and I. Van Kamp. Characterizing urban areas with good sound quality: Development of a research protocol. *Noise and Health*, 16(73):380, 2014.

[235] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, et al. Deep high-resolution representation learning for visual recognition. *arXiv*, 2019.

[236] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2019.

[237] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024.

[238] Y. Wang, N. J. Bryan, J. Salamon, M. Cartwright, and J. P. Bello. Who calls the shots? Rethinking few-shot learning for audio. In *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 36–40. IEEE, 2021.

[239] Y. Wang, D. Liu, and J. Luo. Identification and improvement of hazard scenarios in non-motorized transportation using multiple deep learning and street view images. *Int. J. Environ. Res. Public Health*, 19(21):14054, 2022.

[240] Y. Wang, A. E. M. Mendez, M. Cartwright, and J. P. Bello. Active learning for efficient audio annotation and classification with a large amount of unlabeled data. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 880–884. IEEE, 2019.

[241] Z. Wang, M. Lu, X. Yuan, J. Zhang, and H. Van De Wetering. Visual traffic jam analysis based on trajectory data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2159–2168, 2013.

[242] F. Warburg, S. Hauberg, M. López-Antequera, P. Gargallo, Y. Kuang, and J. Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2623–2632, Seattle, WA, USA, June 2020.

[243] Washington Square Park Eco Projects. Explore birds, 2021.

[244] H. L. Wells, L. A. McClure, B. E. Porter, and D. C. Schwebel. Distracted pedestrian behavior on two urban college campuses. *Journal of Community Health*, 43(1):96–102, July 2017.

[245] K. Wilkinghoff. On open-set classification with L3-Net embeddings for machine listening applications. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 800–804. IEEE, 2021.

[246] D. S. Williams and A. E. Martin. Gait modification when decreasing double support percentage. *Journal of Biomechanics*, 92:76–83, July 2019.

[247] World Health Organization. Global status report on road safety. 2018. accessed on 31 January 2023.

[248] L. Wyse. Audio spectrogram representations for processing with convolutional neural networks. *arXiv preprint ID:1706.09559*, 2017.

[249] F. Xiao, Y. J. Lee, K. Grauman, J. Malik, and C. Feichtenhofer. Audiovisual slowfast networks for video recognition. *arXiv preprint arXiv:2001.08740*, 2020.

[250] J. Xie and M. Zhu. Handcrafted features and late fusion with deep learning for bird sound classification. *Ecological Informatics*, 52:74–81, 2019.

[251] Y. Xu, W. Yan, H. Sun, G. Yang, and J. Luo. Centerface: Joint face detection and alignment using face as point. *arXiv*, 2019.

[252] F. Xue, G. Zhuo, Z. Huang, W. Fu, Z. Wu, and M.-H. Y. Jr. Toward hierarchical self-supervised monocular absolute depth estimation for autonomous driving applications. *arXiv*, 2020.

[253] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024.

[254] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao. Depth anything v2. *arXiv:2406.09414*, 2024.

[255] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How Transferable Are Features in Deep Neural Networks? In *Procedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3320–3328. MIT Press, 2014.

[256] C. Yu, B. Xiao, C. Gao, L. Yuan, L. Zhang, N. Sang, and J. Wang. Lite-hrnet: A lightweight high-resolution network. In *CVPR*, 2021.

[257] L. Yu, W. Wu, X. Li, G. Li, W. S. Ng, S.-K. Ng, Z. Huang, A. Arunan, and H. M. Watt. iviztrans: Interactive visual learning for home and work place detection from massive public transportation data. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 49–56. IEEE, 2015.

[258] J. Zahálka, M. Worring, and J. J. Van Wijk. Ii-20: Intelligent and pragmatic analytic categorization of image collections. *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[259] M. H. Zaki, T. Sayed, and S. E. Ibrahim. Comprehensive safety diagnosis using automated video analysis: Applications to an urban intersection in edmonton, alberta, canada. *Transportation research record*, 2601(1):138–152, 2016.

[260] W. Zeng, C.-W. Fu, S. M. Arisona, A. Erath, and H. Qu. Visualizing mobility of public transportation system. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1833–1842, 2014.

[261] C. Zhang, H. Fan, W. Li, B. Mao, and X. Ding. Automated detecting and placing road objects from street-level images. *Comput. Urban Sci.*, 1:18, 2019.

[262] J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert. Manifold: A Model-Agnostic Framework for Interpretation and Diagnosis of Machine Learning Models. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):364–373, 2019.

[263] J. Zhang, E. Yanli, J. Ma, Y. Zhao, B. Xu, L. Sun, J. Chen, and X. Yuan. Visual analysis of public utility service problems in a metropolis. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1843–1852, 2014.

[264] J. Zhang, M. Zheng, M. Boyd, and E. Ohn-Bar. X-world: Accessibility, vision, and autonomy meet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9762–9771, Montreal, QC, Canada, October 2021.

[265] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang. Bytetrack: Multi-object tracking by associating every detection box. *arXiv*, 2021.

[266] Z. Zhang, P. Webster, V. S. Uren, A. Varga, and F. Ciravegna. Automatically extracting procedural knowledge from instructional texts using natural language processing. In *LREC*, volume 2012, pages 520–527. Citeseer, 2012.

[267] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. *arXiv*, 2016.

[268] T. Zhao, X. Liang, W. Tu, Z. Huang, and F. Biljecki. Sensing urban soundscapes from street view imagery. *Comput. Environ. Urban Syst.*, 99:101915, 2023.

[269] X. S. Zheng, C. Foucault, P. Matos da Silva, S. Dasari, T. Yang, and S. Goose. Eye-wearable technology for machine maintenance: Effects of display position and hands-free operation. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 2125–2134, 2015.

[270] Y. Zheng, F. Liu, and H.-P. Hsieh. U-air: When urban air quality inference meets big data. In *Procedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1436–1444, 2013.

[271] Y. Zheng, T. Liu, Y. Wang, Y. Zhu, Y. Liu, and E. Chang. Diagnosing new york city's noises with ubiquitous data. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 715–725, 2014.

[272] Y. Zheng, W. Wu, Y. Chen, H. Qu, and L. M. Ni. Visual analytics in urban computing: An overview. *IEEE Transactions on Big Data*, 2(3):276–296, 2016.

[273] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy. Domain generalization in vision: A survey. *arXiv preprint arXiv:2103.02503*, 2021.

[274] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra. Detecting twenty-thousand classes using image-level supervision. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*, pages 350–368. Springer, 2022.