THESIS

ROBUST GESTURE DETECTION FOR MULTIMODAL PROBLEM SOLVING

Submitted by

Hannah G. VanderHoeven

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2024

Master's Committee:

Advisor: Nathaniel Blanchard
Co-Advisor: Nikhil Krishnaswamy

Anne M. Cleary

ABSTRACT

ROBUST GESTURE DETECTION FOR MULTIMODAL PROBLEM SOLVING

Throughout various collaborative problem solving (CPS) tasks, multiple different communicative modalities may be used by participants as they communicate with each other to work towards some goal. The ability to recognize and act on these modalities is vital for a multimodal AI agent to effectively interact with humans in a meaningful way. Potential modalities of interest might include, speech, gesture, action, pose, facial expression, and object positions in three dimensional space. As AI becomes move commonplace in various collaborative environments, there is a lot of potential to use an agent to help support learning, training and understanding of how small groups work together to complete CPS tasks. Designing a well rounded system to best understand small group interactions, multiple different modalities need to be supported.

Gesture is one of many important features to consider in multimodal design. Robust gesture recognition is a key component of multimodal language understanding in addition to human-computer interaction. Most vision based approaches for gesture recognition focus on static standalone gestures that are identifiable in a single video frame. In CPS tasks, more complex gestures made up of multiple "phases" are more likely to exist. For instance deixis, or pointing, as it is used to indicate objects and referents in a scene. In this thesis, I present a novel method for robust gesture detection based on gesture phase semantics. This method is competitive with many state of the art computer vision approaches while being faster to train on annotated data. I also present various applications of this method to utilize pointing detection in a real-world collaborative task, and I discuss the importance of robust gesture detection as an important feature in multimodal agent design in further depth.

ACKNOWLEDGEMENTS

When I initially started my Master's degree I did not consider perusing research, especially working full time and being a distance student. I would like to thank Prof. Nathaniel Blanchard for seeing something in me and inviting me to work with the Vision Lab. I am incredibly grateful for all the experiences and opportunities that have come out of switching to a research based degree. I would also like to thank Prof. Nikhil Krishnaswamy for co-advising me and allowing me to work with the SIGNAL Lab. I am beyond grateful for everything I learned from both of them, and how amazing they have been to work with around my schedule. I have learned a lot about academia, applying engineering skills to a research setting and academic writing.

I would also like to thank the iSAT team, especially Mariah Bradford, Ibrahim Khebour, Changsoo Jung, Nathan Kampbell, Carlos Mabrey, and Jade Collins for their support and time as we collaborated over the last few years. This work would not have been possible without them.

I am also incredibly grateful for my husband, Devin Drenk, and siblings Emma VanderHoeven, Evan VanderHoeven and Rebecca (Nathaniel) Quigley, thank you for being supportive of my work over the last 4 years and allowing me to practice presentations on you all. I would like to specifically thank Emma and Devin for their help editing my thesis. I would like to thank my cats, Cheddar and Fitz for staying up with me on late nights. Finally, I would like to thank my parents Jake and Gaynor VanderHoeven for supporting me my entire life and encouraging me to pursue higher education.

This work was partially supported by the National Science Foundation under a subcontract to Colorado State University on award DRL 2019805. The views expressed are those of the author and do not reflect the official policy or position of the U.S. Government. All errors and mistakes are, of course, the responsibilities of the author.

DEDICATION

For my parents

Jake and Gaynor VanderHoeven

thank you for encouraging me to follow my dreams

everything I am is because of you both

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

When humans participate in small group tasks, especially collaborative problem solving (CPS) tasks, they are very likely to employ multiple different forms of communication as they work with each other to reach a goal. Various forms of communication might include speech, gesture, body language, facial expression, and interactions with objects in the environment, amongst others. As artificial intelligence (AI) becomes more prominent in everyday collaborative settings, like in education and workplaces, there is additional potential for an AI solution that helps support learning and team performance by aiding participants as they work together to solve and learn from various CPS tasks.

There are many different applications for an AI agent with the ability to interpret, learn from, and provide feedback to human participants as they communicate with one another. In the traditional education setting our team envisions an AI based solution that would not replace human educators, but rather aid in the education process by enabling students to collaborate, think, and reason as they work together to reach the intended solution. Additionally, this system could provide feedback to the educator on the communication styles and thought processes of individual students. Much like human-to-human communication, for an agent to effectively interpret the current state of any given task, many different communicative modalities need to be extracted and linked together at once. For example, relying on speech only might not give enough context for an agent or human to interpret the indented referent of a statement. Combining speech with another modality, such as gesture, provides more context which in term helps an agent determine the referent of a given statement. Because of this, many design decisions need to be made in order to extract relevant communicative features efficiently, and bring them together in the most meaningful way. This information could be used to extract context from a scene and provide feedback to participants in a communication style that best suits their learning needs.

Gesture is an important and central feature of interpersonal communication. The ability to accurately recognize gestures of interest is vital for an agent to effectively interpret communication in real time. Not only is this crucial to creating multimodal design [29], it is also a critical component of intelligent systems that communicate multimodally with humans [23–25]. Most vision based approaches for gesture recognition focus on static stand alone gestures that are identifiable in a single frame, as opposed to more complex multi-frame gestures. Additionally, most technical solutions for gesture recognition usually rely on large neutral networks that require large volumes of data to train. In my work, I present a 3 stage gesture detection pipeline with the goal of detecting complex multi-frame gestures. In addition, I work to mitigate the mentioned issues and create a solution that returns key frames for a wide range of gestures that could be used in a multimodal system.

Deictic gestures, or gestures that indicate locations in a scene like pointing, are very common in small group communication and add additional context to speech, by allowing speakers to indicate the referent or target of interest in their environment. Leveraging the pipeline in this work to identify key frames for pointing gestures, I extract frames of interest from a video stream, and preform meaningful target detection for use in the overall multimodal system. Additionally, target detection outputs could be combined with other features, like speech, to help add context to statements and help the system understand how humans are communicating as they work to complete the task. Due to accuracy issues involved with using a single plane pointing vector for target detection, since it is unlikely that the objects and vector line up perfectly, I experiment with a "pointing frustum" around the pointing vector to create a "detection" region in three-dimensional space. Objects that intersect with this region, based on the center of the object, are selected as targets of interest [22].

In this thesis, I explore several potential communicative features of interest, as they have been discussed and evaluated by our team, and the design considerations we have made to create an effective agent that could be used in various collaborative settings. Additionally, I go into detail concerning gesture as a feature of interest and discuss methods I have developed for effective, robust and reusable feature extraction in the context of a CPS task. In one such example, I have utilized

the gesture detection baseline to experiment with meaningful object selection using a "pointing frustum." Finally, I discuss how my work with gesture is a vital piece to the overall system, but there are context limitations that come up when relying on a singular modality. These issues could be mitigated in future work by combining outputs with other modalities to bridge context gaps.

Chapter 2

Related Work

Gesture is a common form of nonverbal communication used to convey additional meaning in discussions between persons, small group tasks, and more recently, as a means for humans and computers to interact. In various human computer interaction studies, pointing is a common gesture used to indicate the intended target of a user or study participant. Use of various complex gestures, such as pointing for deixis, spans many different languages and cultures [20], making it an ideal gesture to be integrated with human-computer systems. Pointing is also an important feature of small group communication especially when combined with other modalities, such as speech, as it allows individuals to ground their utterances to the physical environment around them, which adds critical context. For example, any use of demonstratives ("this one," "those," etc.) to refer to physical entities must almost necessarily be coupled with a deictic gesture to be interpretable.

The ability to identify and act on complex gestures in real time is vital in order to utilize gesture as a feature in a larger multimodal system. Correct automated identification of key frames in real time helps reduce noise and aids in accurate identification of gestures, without requiring manually sifting through video data to identify when a gesture starts and stops. In my work, I propose an approach to automatically identify the key phases of gestures to aid in semi-automatic detection and annotation of gesture semantics. Leveraging previous literature on gesture semantics originally theorized by Kendon [17] and elaborated by McNeill [32], and Lascarides and Stone [26], among others. I focus on automatic identification of key frames for gestures that comprise of a *pre-stroke*, *stroke*, and *post-stroke* phases as it unfolds across multiple frames, with the goal of effectively returning the most distinct key frames for identifying any gesture of interest and allowing for versatility when used by an AI agent.

While the ability to identify when an individual is pointing adds useful context to communication, relying only on non-verbal deictic gesture, such as pointing, does not always guarantee accurate target selection. Various experiments have been run to determine the potential increase

in accuracy of pointing when combined with other features, such as speech [14]. In the mentioned study, researchers tested the effectiveness of single plane pointing in an augmented reality from various perspectives, with and without speech. Participants were required to either point at or "identify" (with pointing and speech) an intended target from the various perspectives. They found that combining speech and gesture, increased the accuracy, however there were still errors selecting the intended target.

By utilizing gesture and target detection amongst other features, a multimodal agent can be designed to bring these features together in a meaningful way to extract context from small group tasks. One style of small group work that employs multiple communicative features is collaborative problem solving or (CPS). CPS involves two or more people using "their knowledge and skills to solve complex problems without predefined solutions" [39]. As such, this is a particular method of modeling interaction between users based on research in the learning sciences. Frameworks for CPS have been developed to capture relevant behaviors and different types of collaboration [1, 13, 38]. These frameworks are helpful in creating labeled data and provide a bridge for computer scientists to operationalize and apply knowledge from the learning sciences. Previous work has successfully detected and classified these facets, showed improvements when using multimodal models [4, 37], explored technical requirements on an AI agent for tracking collaboration in small groups, such as relevant toolkits [8], and showed how adding contextual features to models improves the generalizability of the models in collaborative contexts [9]. In this thesis, I discuss features of interest in context of a CPS task in more depth and go into additional details concerning methods I've developed for gesture extraction and real world applications in a multimodal system designed to interpret CPS tasks.

2.1 Datasets

Throughout my experiments involving complex gesture detection, I worked with a few different datasets internal to Colorado State University. In this section I briefly discuss both.

Microgesture Dataset

Microgestures are a category of small, subtle hand gestures requiring less gross motion [43]. The overall goal of such gestures is to reduce the fatigue of a user interacting with a system long-term. The "Microgesture dataset" is a collection of short single-gesture videos, focusing on a single hand. 49 gestures are included in this dataset, each unique and identifiable, fitting into categories including: taps, rotations, move, snap, numbers, zoom, open/close, and slide. Using 3 Microsoft Azure Kinect cameras positioned at 3 different angles, 72 videos were collected for each gesture, spanning 10 participants [15]. Each video consists of approximately 60-81 frames with different start and end points for each gesture, thus making the data a good candidate to test our phase detection solution.

Weights Task Dataset

The Weights Task Dataset (WTD) provides data that can be used to test our gesture phase segmentation pipeline on a more realistic scenario. This dataset is a collection of audio-visual recordings where triads perform a task involving correctly identifying the weights of various colored blocks using a balance scale. Unknown to the participants, the weights of the blocks follow a specific pattern (the Fibonacci sequence), and the task involves participants collaboratively uncovering this pattern. This dataset, which was created to model an example of a CPS task, involves multimodal communication using speech, gesture, gaze, and action in context, making the identification of key gesture semantics important for automated analysis. The dataset spans 10 groups of 3, and includes videos from 3 different camera angles [4, 18]. The videos include many potential gestures of interest, including pointing, grasping, and placing blocks on a scale, which are good candidates for automatic phase segmentation and annotation, in addition to pointing and target detection.

Chapter 3

Multimodal CPS

3.1 Collaborative Problem Solving

In multimodal communicative group work, collaborative problem solving (CPS) is a common method of collaboration in which small groups work together to solve a non-routine task with no set plan. The quality of the group's solution can be evaluated by the team members as the task proceeds, and there is a differentiation of roles but interdependence within the team [13].

A general CPS support agent requires a framework that is not biased to features of a single communicative feature, but can also ground CPS skill indicators to specific events as the task unfolds [4]. To this end, our team has leveraged the framework developed by Sun et al. [38] to categorize our tasks [42]. In this framework, CPS was formalized into hierarchical levels; 19 indicators that include moves such as proposing a correct solution or interrupting others, and three facets which are *Constructing shared understanding*, *Negotiation/Coordination* and *Maintaining team function*. These indicators allow us to identify specific collaborative moves; for example, in Figure 3.1, the participants are discussing the results of weighing two blocks, where *discussing results* is an indicator enumerated in the Sun et al. CPS framework.

Figure 3.1: Stills from the Weights Task Dataset.

3.2 Features

3.2.1 Speech

Speech is a critical method of communication seen in group work. Participants use this modality to share their understanding, ask questions, discuss results, plan, and more. This is an explicit method of communication, making it a foundational starting point for any agent tracking group work. Previous studies demonstrated that speech is a meaningful feature for a model tracking group states [4, 37]. When combined with other features, utterances help add context to how a participant is interacting with the space around them.

In everyday interpersonal communication, speakers are likely to employ demonstrative terms and anaphors ("this", "that", "it", etc.) in place of the specific objects being discussed. Automatically interpreting the intended referent of a statement likely involves context from another modality, potentially deictic gesture. In human-to-human communication it is easy to interpret both gesture and speech at the same time to extract the intended context of any given statement. An AI tasked with understanding how a small group is communicating and working together needs to be able to do the same. For instance, in Figure 3.1, a participant makes the statement "By touch feels lighter," and the utterance alone does not provide enough context to infer which block they are referring to. Perhaps the missing context could be extracted from gesture, specifically the grasp gesture as they place the block on the scale. Because of this link, design considerations concerning the extraction and combination of speech and gesture are vital in creating the most effective agent, and thus is an important line of research which my gesture recognition work greatly contributes to.

3.2.2 Acoustics

Acoustic features convey additional meaning in language, and can be thought of as a subset of the speech feature. From turn-taking to posing a question, the way someone presents their statements provides additional information to others. Acoustic information allows us to understand sarcasm, perceive tone, recognize high energy, and more. For an agent, acoustics (cadence,

prosody, etc.) help classify the sentiment of statements. For example, if someone was to make the statement "It seems pretty balanced" in regards to a scale, but in a sarcastic tone, it would effectively negate the statement. Focusing on utterances alone might lose this additional meaning, thus it's informative to extract acoustics in conjunction with speech. Prior work has shown that acoustics are useful features for a model classifying a group's state [4, 37], thus showing the importance of considering this feature when designing an overall communicative agent.

3.2.3 Facial Expression

Facial expression is an informative modality for an agent tracking group work, as it helps indicate level of engagement, by determining the focus of a participant, perhaps utilizing gaze. Additionally, facial features help determine participants' attitudes towards individual events or even each other. Recent work has shown improvement in facial expression recognition through improvements in deep learning [27, 28]. However, there is a great diversity of ways to interpret different facial expressions, often depending on context. For our purposes I care about the relations of specific expressions to collaborative problem solving. Toward identifying these affects, D'mello and Graesser defined patterns in affective states specific to learning [11]. This allows us to narrow down into expressions that will be important for an agent to track. Some examples of facial expression that might be informative for an agent to understand might include recognizing when a participant is confused and needs extra clarification, or determining if a participant has a positive or negative opinion of something another participant is saying, when coupled with speech.

3.2.4 Physical Space

In order to create an agent that will be able to meaningfully interact with users and support groups, extracting context from physical space helps make inferences about how participants are working together and the relationships between the objects themselves [34]. Leveraging this information and associating it with other features, like gesture, further allows an agent to understand how participants are interacting and the path they are on to complete a task. Mechanistically, object tracking and detection requires common calibration settings to allow data extracted using different

ACTION ANNOTATION

```
(p / put-ACT
  :ARG0 (p1 / participant-1)
  :ARG1 (gb / green-block)
  :ARG2 (o / on
          :op1 (rs / right-scale)))
```

Figure 3.2: Example Action Annotation

tools to be used together (e.g., transform gesture landmarks to the same space as the object locations). The use of 6DOF object pose estimation [10] to extract object locations involves challenges when deployed in group work scenarios, particularly those in classroom environments, where objects in the scene are likely to be small, moved a lot, and subject to partial or complete occlusions. One way to address this is with a model that estimates the object mask to predict the position of an object and then crops the masked image to estimate its rotation.

3.2.5 Actions

Actions in context provide important information about the methods participants are using to compete the task. In a sense, actions can be thought of as an extension of gesture and body language. For example, detecting a grasp gesture signals the potential start of an action that involves putting a block on a scale. Additionally, if an agent can interpret the outcome of an action, i.e., if the scale will be balanced after placing the block, an agent could theoretically track the progress of the group and if they are on track to correctly complete the task. For instance, in Figure 3.1, P2 placing the green block on the scale is an indication of *intent* and of what P2 believes the likely results will be based on the *affordances* [12] of the objects involved: in this example, namely, that the scale will end up balanced. For accurate interpretation of actions, the use of a rigorously-defined interaction semantics for gesture, object and action tracking is vital. Annotation of all task-specific actions engaged in by the participants follows an AMR-style syntax, in which statements are broken down into three arguments: a describer, the subject being described, and the description [3]. Use of this syntax introduces the notion of annotating actions in the style of speech and gesture.

3.2.6 Pose Detection

Similarly to gesture, body pose detection can be used as a feature of interest in CPS scenarios. Important context is likely to be associated with gross body motion in addition to fine-grained joint positions on the hand. Using the depth channel from Azure Kinect recordings, the x,y, and z positions of 32 joints across the body are extracted consistently for each participant on a frame by frame basis, much like hand detection with MediaPipe [44]. The tracked body joint positions are evaluated for each participant to make meaningful inferences about how group members are interacting. The general body language of an individual helps determine levels of engagement in a task, in addition to focus and attention. If a participant's body is facing towards others in the group it might signify that they are engaged with what the group is doing or what someone is saying. Additionally large movement might indicate significant changes in overall atmosphere. Because the raw joint positions are anchored to the physical location of the different participants, the individual bodies either need to be segmented and processed individually with distinct models for each person, or transformed into a normalized space before feature processing and classification. Additionally, the joint positions for each participant could be used to help localize other modalities of interest, for example, narrowing down the position of an participants hands for gesture detection or the position for their face to determine gaze vectors and facial expressions for efficiently.

3.2.7 Gesture

Gesture and pose add context to verbal communication, particularly regarding how someone interacts with the space around them. For example, one common salient gesture in the Weights Task is deixis (pointing), and since situated shared tasks often involve use of demonstratives ("this"/"that"), interpreting them involves recourse to deictic gesture. Detecting semantic content of a point gesture helps identify what a participant is paying attention or attempting to draw attention to, allowing the interpretation of communicative features within the physical space [40]. For the purposes of model training and for post-facto interpretation, some high-level semantic representation of gestures, such as GAMR [6] are required. The recognition of gestures themselves

GAMR GESTURE UNIT

```
(g / gesture-unit
  :op1 (i / icon-GA
    :ARG0 (g2 / gesturer)
    :ARG1 (b / block)
    :ARG2 (a / addressee))
  :op2 (d / deixis-GA
    :ARG0 g2
    :ARG1 (l / location)
    :ARG2 a))
```

Figure 3.3: Example GAMR Gesture Unit

is largely based on positions of the individual joints on participants' hands [33]. These joint values are extracted using tools such as MediaPipe [44] and used to calculate the pose of the hand and therefore recognize gestures. Other gesture and pose-based features include distance between subjects, indicating engagement and inter-subject attitudes. Capturing this often requires depth information, necessitating RGBD cameras such as the Azure Kinects used to record the Weights Task Dataset.

GAMR

Gesture AMR (GAMR) is a formalism that encodes the meaning of gesture in multimodal interactions between agents. It is an extension to Abstract Meaning Representation (AMR), adopting both the annotated graph structure and the predicate-argument representation of that formalism [3]. Gesture AMR was developed to encode how gesture meaning both independently of and as it relates to speech.

Gesture AMR distinguishes four general types of referential gestures: *iconic*, *deictic*, *metaphoric*, and *emblematic* [16, 21, 30, 31]. Because our data focuses on gestures in a task-based setting, most depictions of entities and events appear to reflect their concrete properties, such as the shape of an object or the manner of an action. Similar to the interactions reported on in [6], metaphoric gestures do not appear with any frequency.

GAMR includes schemata to annotate gestures that fall into one or more of these categories, thus providing granularity when representing a variety of gestures that might be used to communicate in various CPS tasks. The inset shows the structure for a "gesture unit" including both deixis and iconic components. `ARG0` denotes the gesturer, `ARG1` the semantic content of the gesture and `ARG2` is the addressee or intended recipient; these fields exist for each gesture subsection in the annotation.

Chapter 4

Robust Gesture Detection

4.1 Gesture Detection Pipeline

In order to perform meaningful robust gesture recognition, I developed a novel pipeline that aims to automatically identify the key phases of gestures and aids in semi-automatic annotation of gesture semantics through identifying the sub-gestural phases (see Chapter 2), [17, 26, 32]. Specifically, I focus on the automatic detection of "key frames," which I define as frames comprising the union of the *hold* phases of any given gesture, specifically a combination of the pre-stroke, stroke, and post-stroke phases. I theorize that the *hold* phases make up the majority of the semantically significant movement of a gesture. My gesture recognition pipeline consists of three stages that aid in reducing noise and distilling videos down to these key frames. It has been evaluated over a collection of complex, multi-frame gestures in two datasets (see Section. 2.1). Key components of the pipeline include a classification model whose main purpose is to recognize the general static shape of complex gestures when in a *hold* phase (Section 4.1.2), a movement segmentation routine which aids in breaking down a video into segments of similar movements (Section 4.1.3), and a phase markup annotation which uses the classification model and the video segments to identify and annotate the segments and frames that are in a *hold* phase, and thus most semantically significant or adjacent to the most semantically-significant frames (Section 4.1.4). Figure 4.1 shows the gesture detection pipeline, with the addition of our steps taken for point based target detection (see Chapter. 5). In this section I will walk step by step through each piece of the pipeline and tools used in more detail.

4.1.1 MediaPipe

Hand detection tools, like MediaPipe, an open source library developed by Google [44], support gesture recognition tasks by detecting and returning joint locations of individual hands in a

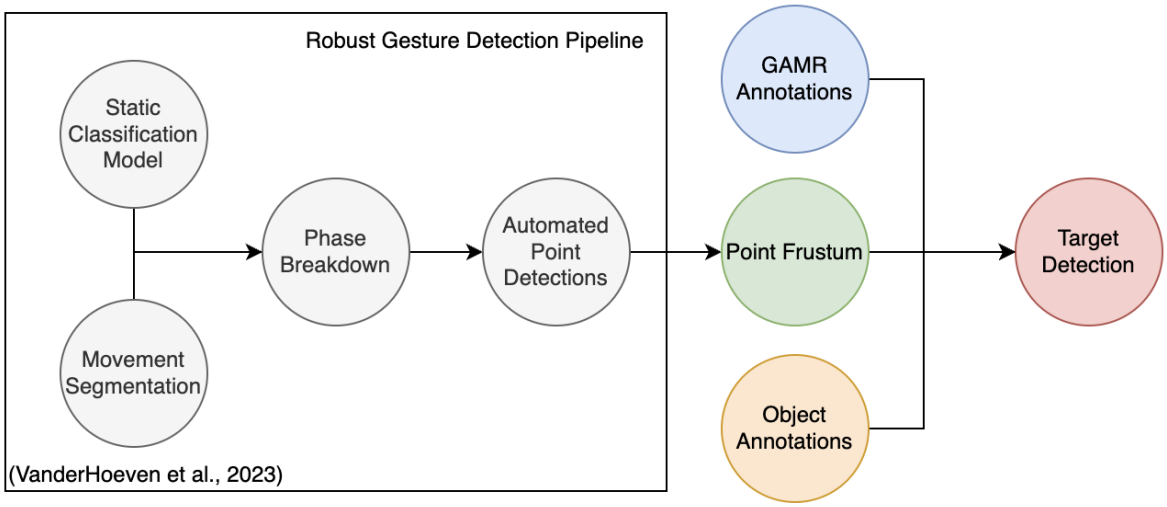


Figure 4.1: Complex gesture and target detection pipeline. Items within the box denote components of the robust gesture detection pipeline (discussed in Chapter. 4), the values on the right are outline how the outputs of the gesture pipeline have been used for meaning point target detection (discussed in Chapter. 5).

frame. MediaPipe tracks hands using 21 landmarks that consist of x, y, and z (relative depth) coordinates. It uses a two-stage pipeline: 1) the palm, as opposed to the full hand with all fingers, is detected using a neural model over the full input image. The palm based bounding box is returned and formed to mark its initial location; 2) more precise hand landmarks (joint positions) are located using encoder/decoder feature extraction based on the location of the palm bounding box. Figure 4.2, shows the constant joint locations or *landmarks* of detected hands, meaning index 0 is always the "WRIST", index 4 is always the "THUMB_TIP". These landmarks values are used to train custom gesture recognition models that can be used for a variety of tasks. Additionally, models are trained more efficiently than counterparts using RGB data, by removing noise from RGB data and focusing solely on the shape of the hand.

4.1.2 Static Classification Model

The first step of my gesture detection pipeline is creating a binary classifier whose purpose is to help identify individual frames where the hand appears in the general shape of the "stroke" phase

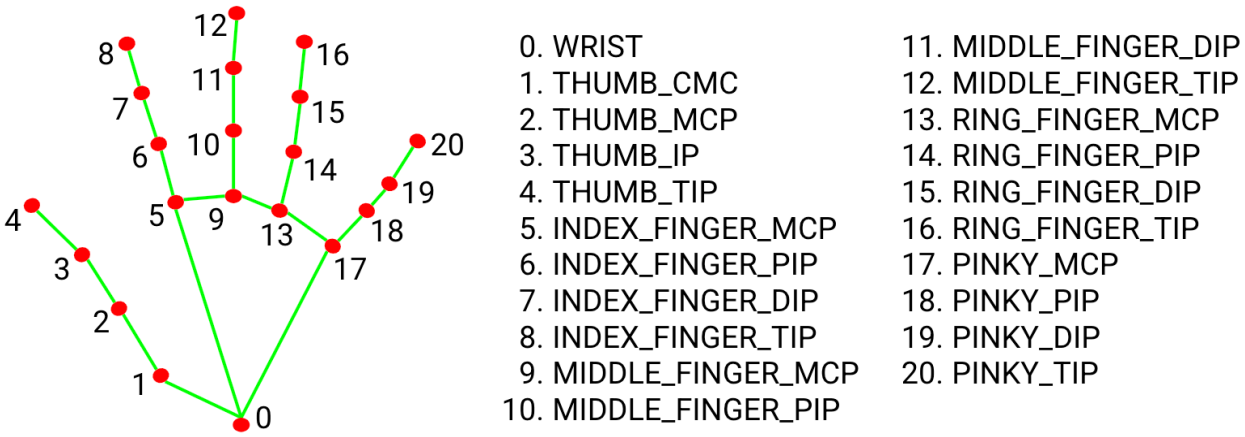


Figure 4.2: MediaPipe Hand Landmarks (reproduced from [44])

of a given gesture of interest. In this section I outline the annotation method used to create data for this classifier and the details of the classifier itself.

Annotation

Some manual annotation is required to create a static classification tool for any given complex gesture. In order to create this data, I developed an annotation script for the Microgesture and Weights Task datasets. Using this script I programmatically step frame by frame through a sample video using OpenCV, and for each frame and identifiable hand I save off the *phase type*, *gesture type*, *participant ID*, and normalized landmarks returned from MediaPipe. For the Weights Task dataset I include additional fields: *hand index* and *group ID*. *Phase type* can be one of two values: *hold*, which signifies a frame in which the hand shape is similar to the shape of the gesture in any of the stroke phases, and *no hold* which represents a noise shape (dissimilar from stroke) that should be ignored. *Gesture type* maps directly to a user defined index of the gesture. Figures 4.3 and 4.4 show examples of annotated data from each of the datasets used in this work.

For annotation using the Microgesture dataset, I used the default tracking and detection values for MediaPipe (cf. Section 4.1.1), and set the maximum number of hands to 1. For the Weights Task data both tracking and detection values were set to 0.6 with maximum hands set to 6, thus allowing us to annotate any combination of hands returned from MediaPipe.

Figure 4.3: Annotation examples from Microgesture dataset

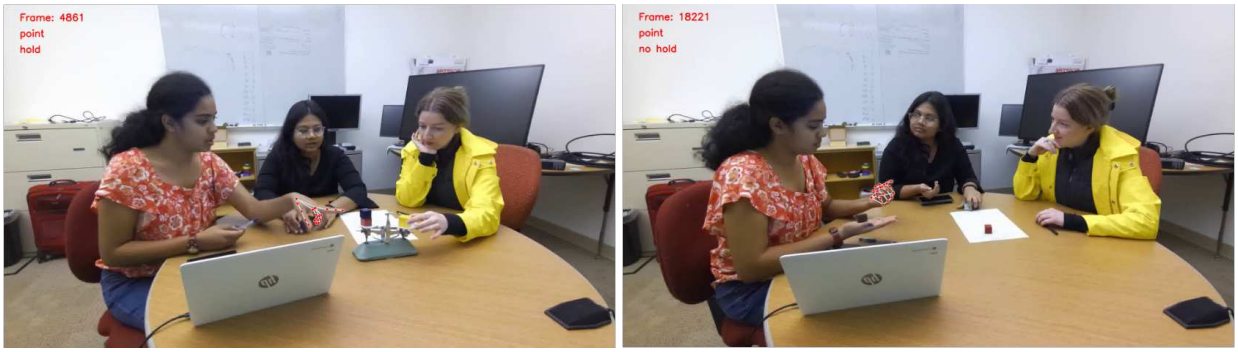


Figure 4.4: Annotation examples from Weights Task dataset

Classification Model

To test my pipeline I experimented with a subset of gestures from the Microgesture dataset and created a classification model for each. To generate this subset I selected one gesture from a few of the hierarchical categories of the Microgesture dataset [15]. I selected one gesture from the following categories: *move*, *snap*, *number*, *zoom*, *open/close*. The categories were chosen with the intent to create a diverse selection of gestures across multiple categories. I selected one specific gesture from each category, including *two* for number, *index finger swipe right* for move, *hand close* for open/close, *zoom in with palm* for zoom and *snap* for snap. Because each gesture comes from a different category, I expected each to be distinctly identifiable using key frames. Fig. 4.5 shows samples of each of the 5 gestures of interest selected for evaluation.

Using the annotated values for each frame in the sample data for each gesture, I created a random forest binary classifier using the `scikit-learn` library for each gesture of interest. A random forest classifier is made up of a collection of independent decision-tree classifiers generated from a randomly selected subset of training data. These classifiers then vote on the most probable

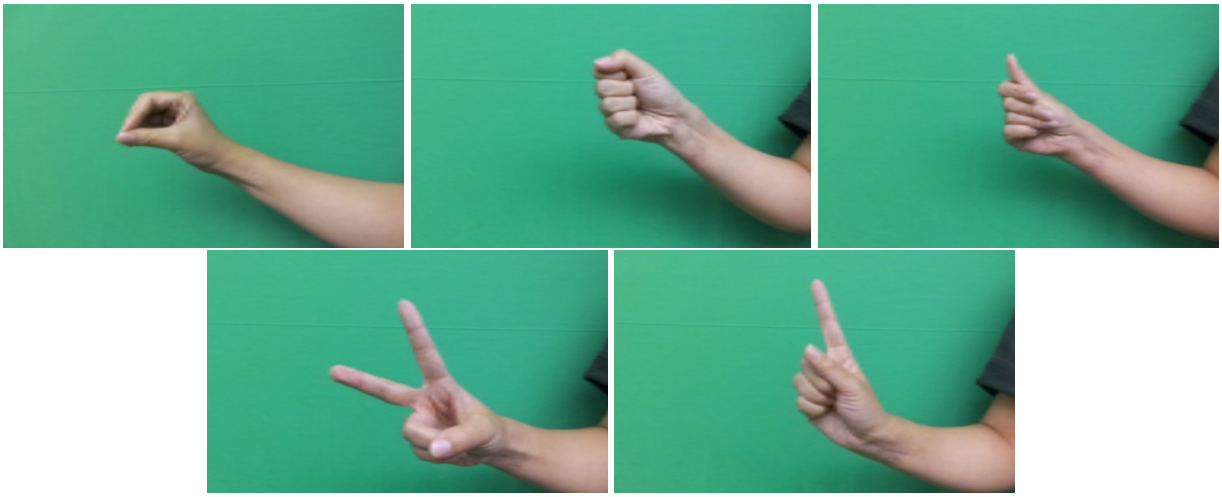


Figure 4.5: Hold phase stills (from top left): *zoom in with palm*, *hand close*, *snap*, *two*, and *index finger swipe right*

Table 4.1: Overall accuracy, balanced accuracy, Kappa, and AUROC. Gestures of interest come from the Microgesture Dataset.

Gesture	Accuracy	Balanced Accuracy	Kappa	AUROC
Two	0.92	0.93	0.92	0.92
Index finger swipe right	0.91	0.87	0.89	0.89
Hand close	0.87	0.78	0.84	0.82
Zoom in with palm	0.83	0.91	0.88	0.87
Snap	0.94	0.95	0.95	0.95

classification for any given input [5]. Random forest classifiers have been used in other MediaPipe based classification projects (e.g., [7] and [35]). The overall purpose of this classification model is to identify when a hand is in a *hold* phase or not. Table 4.1 shows the accuracy metrics, Cohen's Kappa, and AUROC values, which were selected as metrics to show the overall performance of a binary classifier beyond just using accuracy, of each of the random forest classifiers generated. The high values for all metrics on each gesture shows that landmarks can be used to identify the general shape of the *hold* in a single frame; however, as mentioned before this is not enough to identify key frames and additional processing is required.

4.1.3 Movement Segmentation

The pipeline's second phase involves grouping similar individual frames into like movements. For each frame, I computed a value representing the hand's general location in the frame, specifically the average (x, y) values in pixel space, using the 21 landmarks returned from MediaPipe. Fig. 4.6 shows an example of average hand location determined from the individual landmarks. Hereafter I defined a collection of frames with a difference in average location under a defined threshold as a "segment."

In order to generate segments for a video, I first cached off information for each frame, including the video number, frame number, MediaPipe landmarks, the normalized landmark values, and the average (x, y) location of the hand. If the current frame's average location is significantly different from the last frame this marks the start of a new segment, otherwise the frame is added to the current segment. The threshold value used in our experiments is 0.8 pixels, however the most effective value for other scenarios or datasets may very and thus the value is user-definable. Figs. 4.7 and 4.8 show a visualized example of these steps on a subsection of *zoom in with palm* frames.

4.1.4 Phase Breakdown

After the video has been broken down into segments, I analyzed the segments using our classification model to identify segments with a significant percentage of frames in *hold*. For each segment in the video I ran the static shape classifier on each frame in the corresponding segment. If at least 80% of the frames are in *hold*, I mark that entire segment in *hold*. During this process I look for the following pattern: the first *hold* segment found marks the potential start of the key frames interval, or a "section of interest." After a *hold* is found, the next *no hold* segment marks the end of the section of interest. The entire detected section should span the pre-stroke, stroke, and post-stroke phases. For each section of interest found in a video, I also calculate the total number of frames. The user can define the number of frames required to mark the section as significant or

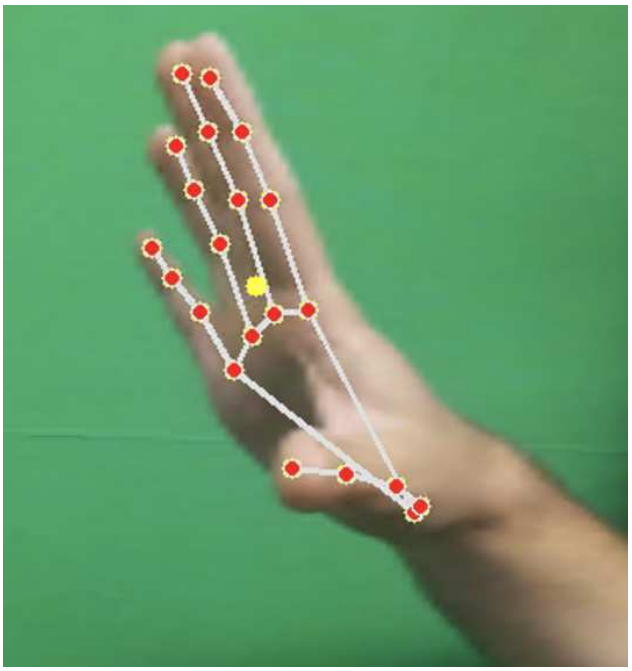


Figure 4.6: Example of average hand location, shown by the yellow dot.

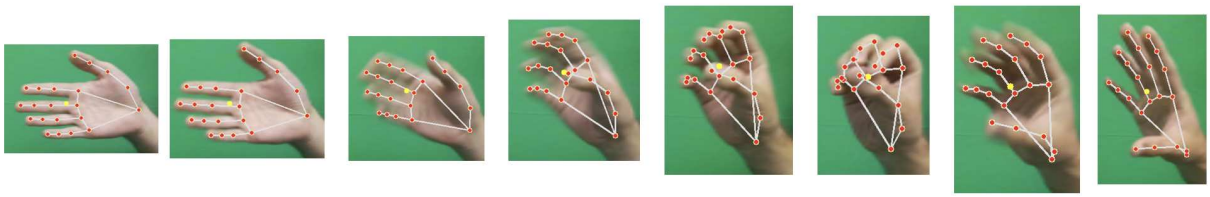


Figure 4.7: Movement Segmentation: cached frame values

key frames and thus as a candidate for automatic annotation. Fig. 4.9 shows a visualized example of this grouping, continuing the example from the previous section.

Once key frames are identified, a few values are saved off that can then be utilized to help extract key frames from a video subsequently to train additional models to classify a collection of gestures in real time. These values include the first and last frame of the key frame section, and the first and last frame of the peak segment, which is defined as the segment that contained the most frames. In addition, the frame count for each segment can be plotted to show the pattern of motion for any gesture of interest. It is worth noting that the pattern of peaks and valleys on this chart can be used to map the occurrence of the stroke phase, and for some gestures this could take place in or around the peak segment. Figs. 4.10-4.12 show example plots of the distributions for

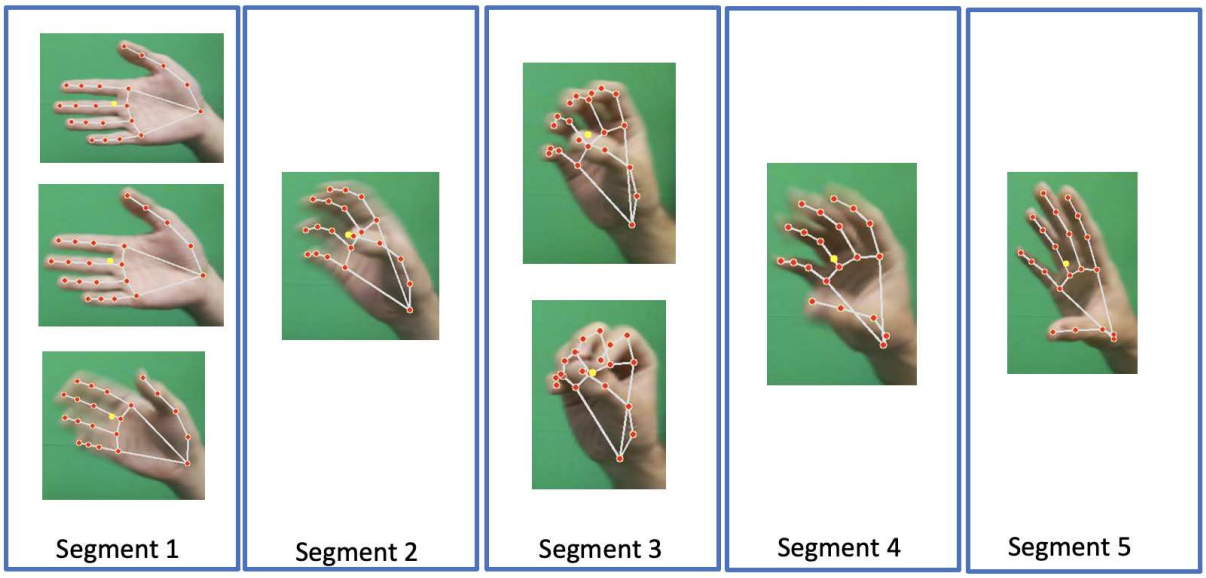


Figure 4.8: Movement Segmentation: creating segments

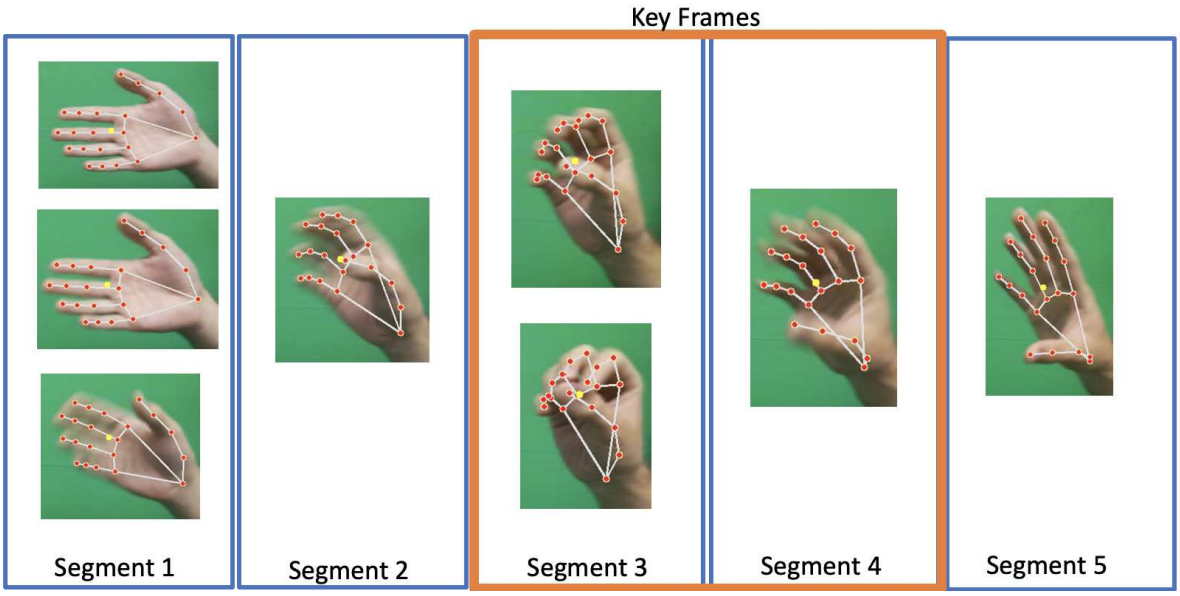


Figure 4.9: Phase Breakdown: locating key frames

each of the five gestures examined. It is worth noting that for each gesture the start location and length of the key frames vary, showing the importance of dynamic key frame selection. Selecting one static key frame location across all values in the dataset would lead to noise in training inputs. The "peaks" shown in these charts indicate a section of frames where motion was relatively small

(that is, a high concentration of similar frames), whereas valleys show locations where change in motion is more significant.

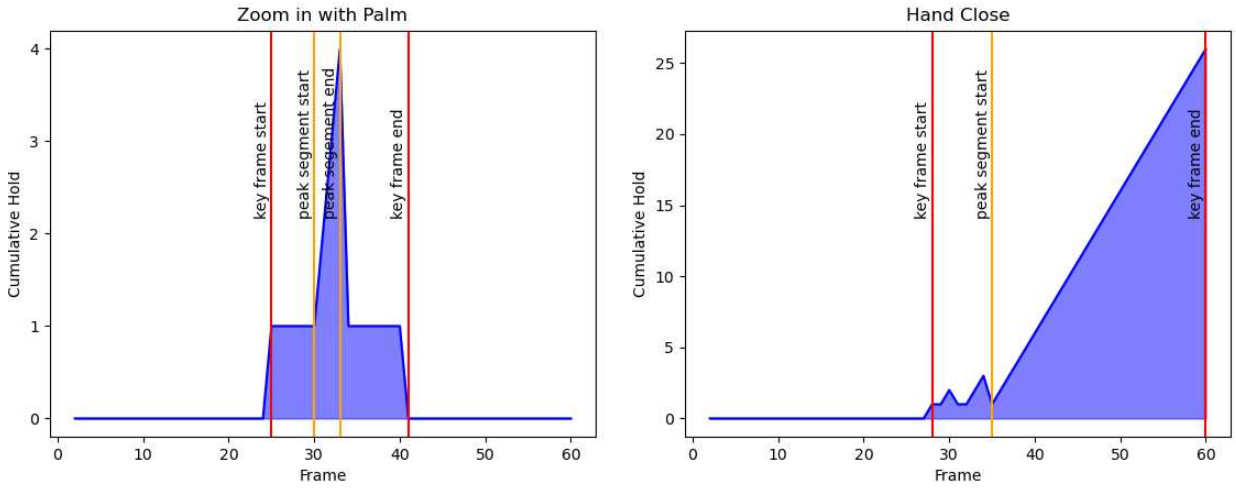


Figure 4.10: Key frame examples from *zoom in with palm* and *hand close*

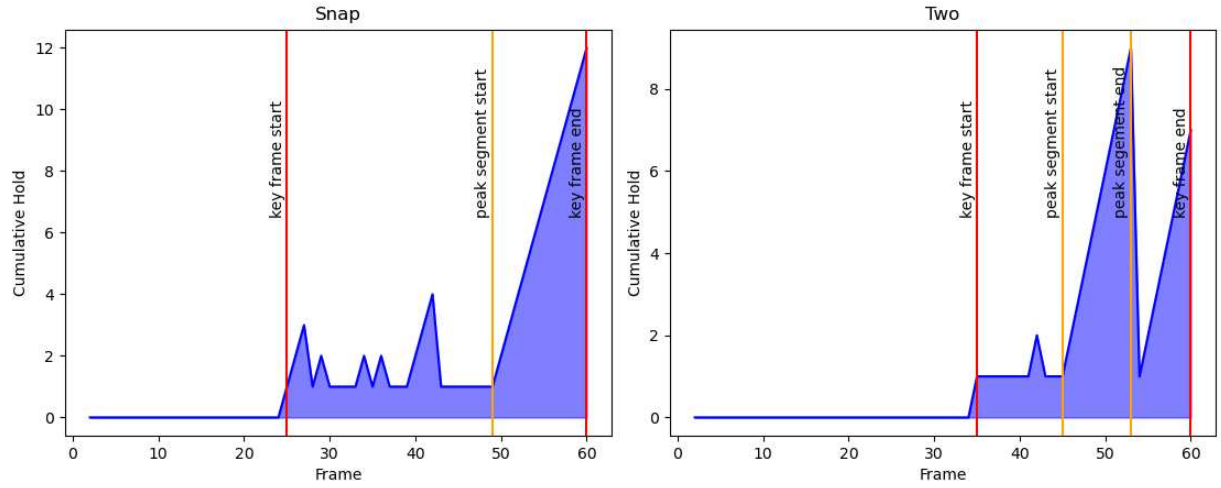


Figure 4.11: Key frame examples from *snap* and *two*

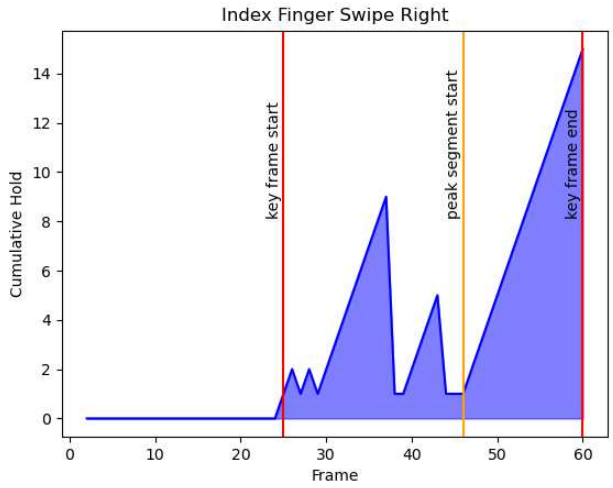


Figure 4.12: Key frame examples from *index finger swipe right*

4.2 Experiments and Evaluation

The major motivation in this work has been to develop a way to automatically pinpoint the exact locations of key frames for any given gesture, so they can be utilized in a larger multimodal system. Success in this aim means that automatically extracted key frames should be more informative of a gesture to a recognition system than a naive or random baseline. In my work I've demonstrated success on such a recognition task.

My initial goal was to train a classifier to recognize multi-frame gestures across the entire Microgesture dataset using MediaPipe's landmark data. In an initial attempt, where all frames in each video were gathered and fed into a simple classifier, the resulting accuracy that was equivalent to a random guess. To reduce noise and trim down the number of frames being fed into our classifier, I tried gathering 10 frames starting at the 20th frame for each video. This resulted in performance very similar to the first attempt. Various additional attempts were made to select a number of frames from the same static location in each video that best fit the entire dataset. Through my experiments I was able to improve accuracy slightly only to a maximum accuracy of about 20%. This value using naive or static key frame selection serves as a baseline. I hypothesize this occurred because the true location of the key frames varied greatly for each gesture, and no single starting position would be effective across the entire dataset. Selecting one static location

across all videos leads to noise being mixed into the training data. In order to improve accuracy key frames would need to be identified on a video by video basis, and our dynamic key frame selection method serves as a possible automated solution.

To test this proposed solution I repeated our initial experiment on the Microgesture subset referenced in Section 4.1.2 (the gestures of interest include *snap*, *two*, *index finger swipe right*, *hand close*, and *zoom in with palm*). Using both the static and dynamic key frame collection methods, I trained a feedforward neural network on this subset, using 2 ReLU-activated hidden layers of 20 and 10 units, respectively, with a final softmax classification layer. The model was trained for 100 epochs with an Adam optimizer, a learning rate of 0.001 and batch size of 16, using a leave one out split for each of the 10 participants. In both cases I gathered 10 frames from each video.

The experiment results show an average of all 10 leave-one-out splits. Table 4.2 shows statistics from both classifiers. Table 4.3 shows associated standard deviations across all 10 splits for each figure. The baseline values are much higher than the values in my initial experiments that spanned the entire Microgesture dataset. For the subset selected, the static frame selection does better at selecting relevant information. However, the results show that when using dynamic key frame selection, there is a significant increase in overall performance. Figs. 4.13 and 4.14 show the performance of the classifier using both frame selection methods in the form of confusion matrices (summed over all splits). It is evident from Fig. 4.14 that dynamic key frame selection does significantly better over all the gestures of interest. The experiments show that my pipeline and dynamic key frame selection is a promising solution to reduce visual noise in a dataset by focusing on segmenting gestures using semantically-informed key inflection points, thus improving the performance of models tasked with identifying complex multi-frame gestures, that in turn could be used to help identify movements on interest in various CPS tasks.

Table 4.2: Average reported metrics: precision, recall, F1, and top-k accuracy (selected to show the additional detail of the multinomial classification) 10 frames were gathered for both methods. Static collection started at frame 20 of each video. Dynamic started at the unique key frame location for each individual video.

Method	Precision	Recall	F1	Top-1	Top-3
Static	35.28	39.16	33.86	41.48	83.49
Dynamic	66.35	66.48	62.78	69.10	89.10

Table 4.3: Standard deviation of reported metrics: precision, recall, F1, and top-k accuracy.

Method	Precision	Recall	F1	Top-1	Top-3
Static	23.99	20.36	21.97	21.16	9.78
Dynamic	22.80	19.00	22.00	18.28	9.86

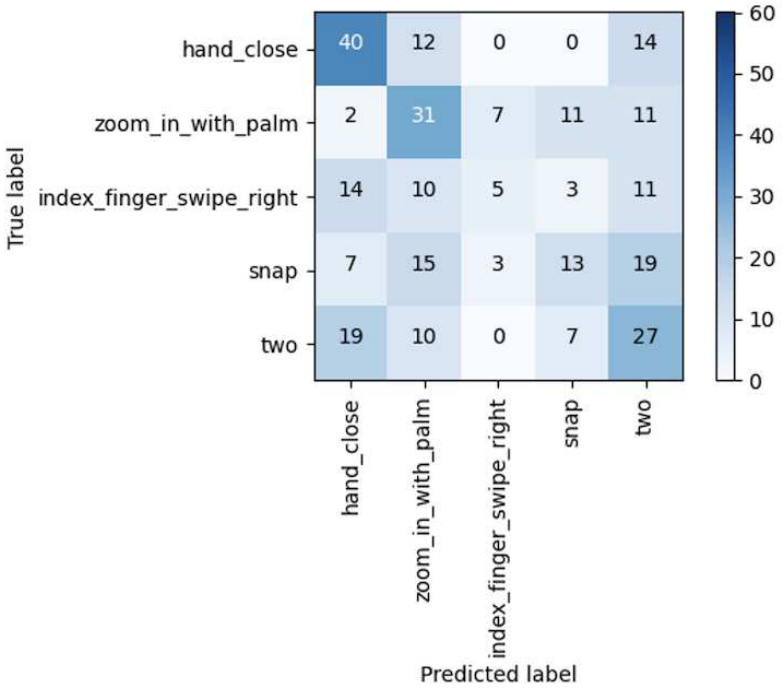


Figure 4.13: Classification results using static key frame selection

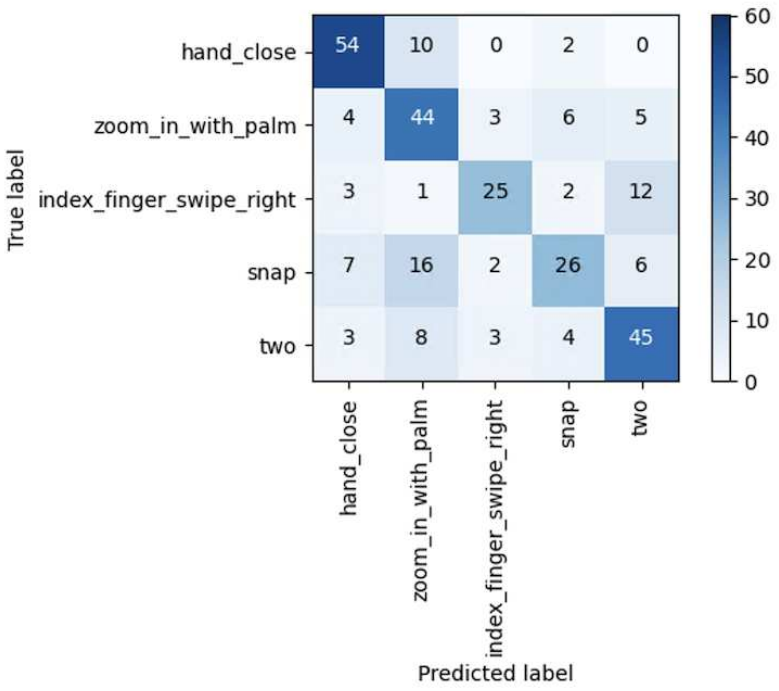


Figure 4.14: Classification results using dynamic key frame selection

Chapter 5

Pointing For Target Detection

A critical component of multimodal human-human interaction is deictic gesture (pointing), and therefore accurate identification of pointing targets in real time is an important feature for multimodal language understanding and human-computer interaction. By using pointing vectors to aid in identifying targets in three-dimensional space, the semantic target of a user is extracted from a video stream [41]. In addition, when combined with other features, such as speech, the data extracted further aids the overall understanding of how humans are communicating with one another or with an intelligent system. For the most accurate analysis and seamless use, correctly and consistently identifying people, gestures, and their intended semantic targets in real time is vital. Leveraging my previously developed pipeline I automatically detect preparatory, "stroke," and recovery phases of gestures [40], based on the gesture semantics previously developed by the community [2, 17, 26]. I have been able to show promising results in automatic identification of complex multi-frame pointing gestures in real time, which in turn could be utilized a the larger multimodal system.

To effectively extract instances of pointing and the associated targets from a small group scenario, a few things must be considered. First, accurate gesture and body detection on a per-participant basis is necessary to consistently match a pointing vector with who is communicating. Second, precision errors can occur when a single vector is used to select objects since it is unlikely that the objects and vector line up perfectly. van der Sluis and Krahmer [36] studied deixis in the context of multimodal referring expressions and found a main effect of distance. The decreased specificity of pointing over distance can be modeled as a “cone” *a la* Kranstedt et al. [22]—a volume narrower at the vertex (the pointing digit) and wider as distance from the digit increases. To account for this, a "pointing frustum" (viz. a cone with the tip truncated) is formed around the pointing vector to create a "detection" region in three-dimensional space. Objects that intersect with this region, based on the center of the object are selected as targets on interest [22].

As pointing specificity degrades with distance from the pointer to the target but is still interpretable by other humans at a distance [36], selecting the most fitting near and far base radii to create the pointing frustum is important to correctly identify intended targets in a small space, without selecting unintended targets. I compare an automatic pointing detection method with a human-annotated ground truth, and frustum radii to determine the feasibility of point and target detection of small objects. With these points in mind, I have established a novel baseline for object detection in a joint situated task using deictic gesture only, and in the process expose how challenging automatic inference of indicated objects in a collaborative setting could be due to variation across individuals and groups in communication and deictic strategies.

5.1 Methodologies

5.1.1 Data Preprocessing

A few data prepossessing steps were required in order to test the proposed target selection solution using the WTD. Human annotation of frames in which pointing gestures occurred were gathered for a subset of groups. This process involved manually stepping through each frame and marking a participant ID[1] along with the start and stop frame for each deictic gesture. For each manually annotated frame the team, saved the block's color, quaternion describing its location, location in 3D Cartesian space, and 2D bounding box information. This gives a maximally precise object location in each frame against which to assess the quality of object selection with automated deixis detection compared to a human-annotated ground truth. From there with the help of the team, we ran a linear interpolation algorithm to fill in the object locations for the intervening video frames. Figure 5.1 shows an example of the target blocks on the scale, with and without the overlaid 2D bounding box drawn from the manual annotations. Because of the time required to annotate each video, only a representative subset of groups were selected for our experiments.

[1]Participants are conventionally indexed 1–3 from left to right in the video frame.

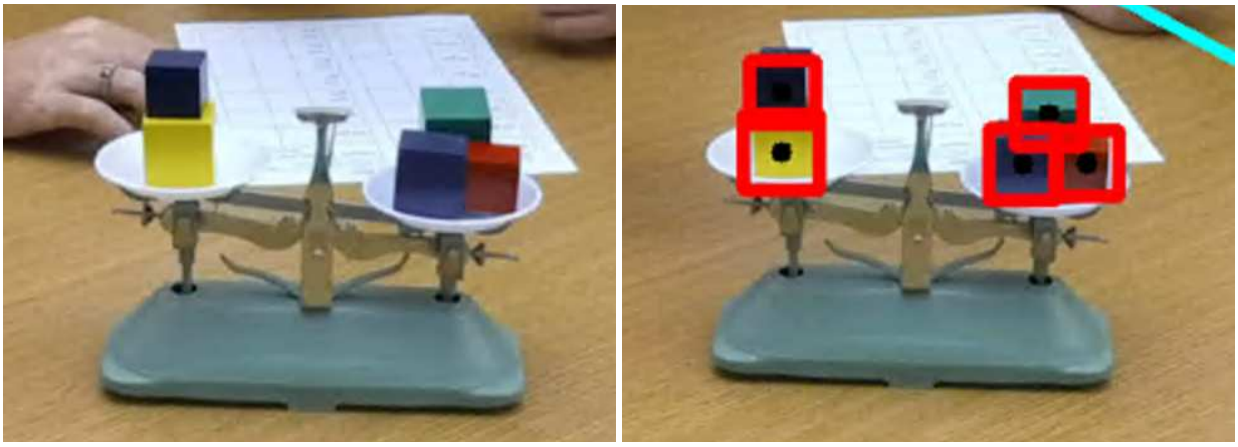


Figure 5.1: Target blocks, with and without bounding box overlay.

5.1.2 Pointing Frustum

Bringing together MediaPipe and our robust gesture recognition pipeline, I automatically identify frames of interest for deictic gestures, and from there use the hand landmarks to calculate a pointing vector to identify target objects in a scene. For the purposes of my experiments I calculate the pointing vector by extending a ray through the base and tip of the index finger, comprising the MediaPipe landmarks at index 5 and 8, respectively. I then extend the vector out into the environment 5 times the distance from finger base joint to fingertip, starting from the tip of the index finger (joint 8). Figure 5.2 shows the joints used to create the pointing vector relative to the MediaPipe landmarks.

When using a vector embedded within a single plane to detect targets of interest, it is very unlikely that the vector and object of interest will line up perfectly. Because of this, I use a "pointing frustum" to create a target detection region. A frustum is a geometric shape resembling a cone, where a radius value is set for the top and bottom of said cone. I specify a "near" (or top) radius at the base of the index finger and "far" (or bottom) radius at the end of the pointing vector to allow increased granularity when experimenting with detection regions. This reproduces the pointing cone semantics of [22], and makes it extensible to allow for different levels of imprecision at distance. Figure 5.3 shows a top down view of the frustum, note that in a real scenario, the frustum is a three-dimensional volume with circular cross-sections. The red block in the figure denotes

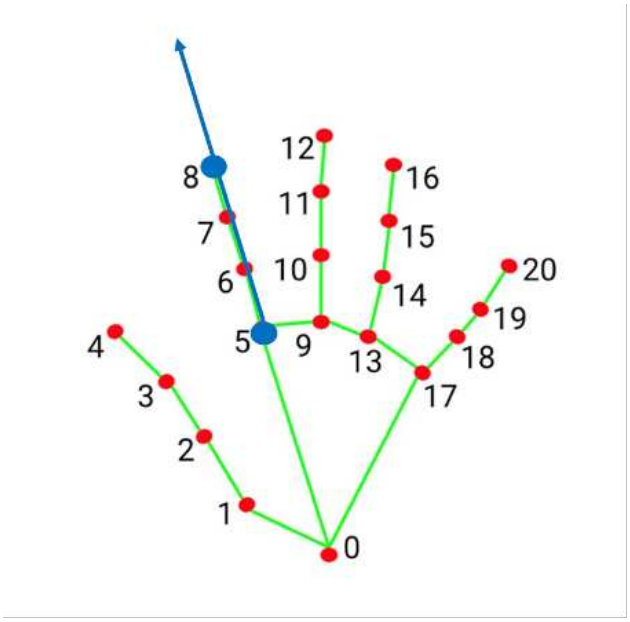


Figure 5.2: Pointing vector relative to MediaPipe landmarks.

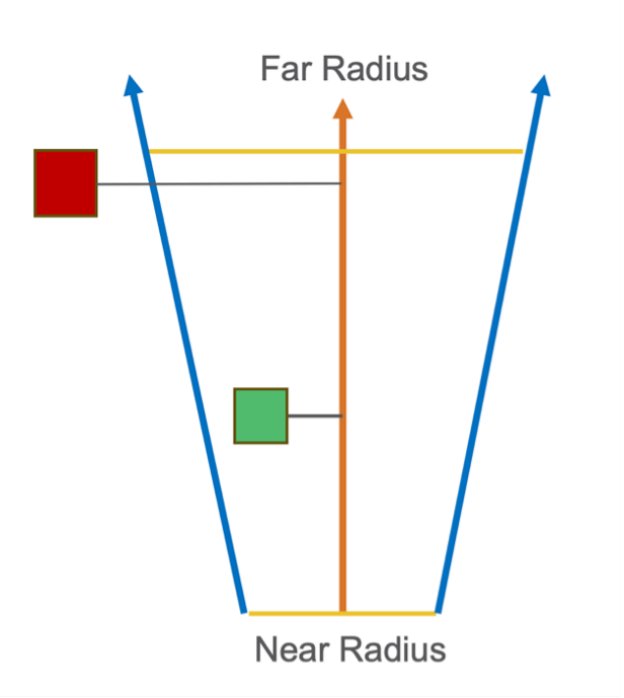


Figure 5.3: Top down view of the pointing frustum. The red block is an example of an object that falls *outside* the detection region, the green block is an example of an object that falls *inside* the detection region and would be marked a target of interest.

a target object outside the detection region, whereas the green box is an example of a target of interest. I determine if a target is in the detection region by first finding the location of the target

$$\vec{P} = p2 - p1$$
$$\vec{B} = p3 - p1$$
$$\vec{A} = \frac{\vec{B} \cdot \vec{P}}{\|\vec{P}\|^2}\vec{P} \qquad (5.1)$$
$$x = p1 + \vec{A}$$

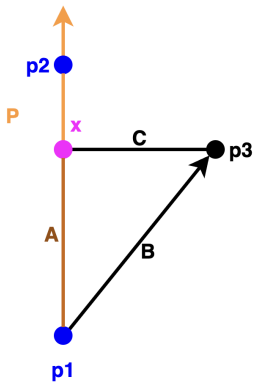


Figure 5.4: Calculation of pointing vector target, where P represents the pointing vector, $p3$ represents the center of a target object, and x represents $p3$ projected onto P

perpendicular to the vector, outlined in Figure 5.4. From there the radius of the frustum is found at that point and it is determined if the center of the object is within that radius. This process is depicted in Figure 5.4 and Equation 5.1. x denotes the center of the candidate target object projected onto the pointing vector (P). $r_d = r_n + \frac{r_f - r_n}{\|\vec{P}\|}\|\vec{A}\|$ gives the radius of the pointing frustum at distance $\|\vec{A}\|$ from the "near" plane (where r_n and r_f are the near and far radii, respectively). If the distance from the center of the candidate target to its projection x is less than or equal to r_d, the candidate lies within the pointing frustum and is considered "retrieved."

5.1.3 Azure Landmarks

In order to implement gesture recognition on multiple participants, additional assurances needed to be built on top of the MediaPipe hand recognition [40]. MediaPipe includes the ability to track multiple hands but does not guarantee the order of the returned hands. This means that it is possible for participants' hands to get mixed up (for instance, if they overlap, or leave the frame and return), leading to incorrect assignment and therefore gesture classifications and attributions. For instance, if participant 1 is pointing at the blue block, but the gesture is associated with participant 2, the result would be inaccurate representations of gestures within the scene. I therefore took the locations of joints on the bodies of the different participants, which were extracted from the depth video stream (see Figure 5.5). Using these, I calculated a bounding box on each of the partici-

pants' hands, to allow MediaPipe to retrieve the hand joints from a localized area. By tracking the bounding box to the wrist joint according to Azure, I more consistently associate hands (and thus complex gestures and movements) with a participant.

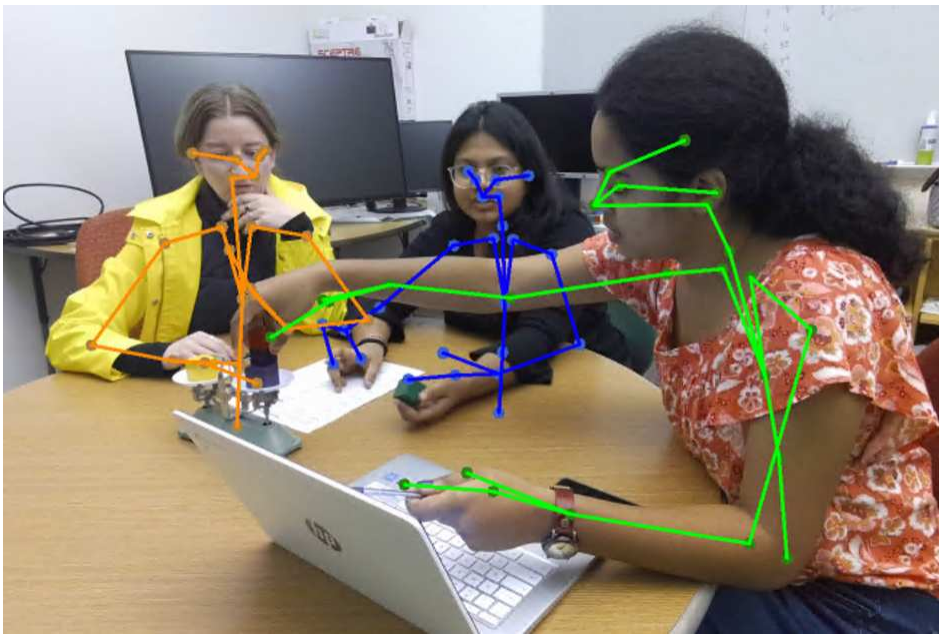


Figure 5.5: Azure body landmarks overlaid on a frame.

5.1.4 Depth Information

In addition to body landmark locations, the Azure SDK facilitates retrieving framewise depth information. This depth information is saved as grayscale Z-coordinate values for each pixel in field of view, measured in millimeters. Figure 5.6 shows an example depth frame with a semi-opaque overlay of the RGB data. The depth information allowed me to convert hand landmarks and object locations between two-dimensional and three-dimensional space, thus allowing us to create a pointing vector, and detect targets in three-dimensional space.

5.1.5 GAMR

Ground truth values for the intended target objects annotations are provided in the WTD in the form of Gesture Abstract Meaning Representation (GAMR) annotations [6]. Refer back to

Figure 5.6: Azure Depth Data with RGB overlay

Section 3.2.7 for additional details on the annotation scheme. For the purposes of the experiments in this chapter, I focus only on the *deictic* gesture type, i.e., gestures that refer to a location by pointing. Figure 5.7 shows an example of a pointing gesture referencing a block. Figures 5.8 and 5.9 show examples of GAMR annotations from the WTD. Note that in, e.g., Figure 5.8, the gesturer (`ARG0`) is `participant_1` and semantic content of the gesture (`ARG1`) is the `blue_block`. This information is used in conjunction with the targets selected by intersection with the pointing frustum to verify if object selections were correct.

```
(d / deixis-GA
  :ARG0 (g / gesturer)
  :ARG1 (b / block)
  :ARG2 (a / addressee))
```

Figure 5.7: Deixis GAMR template according to [6].

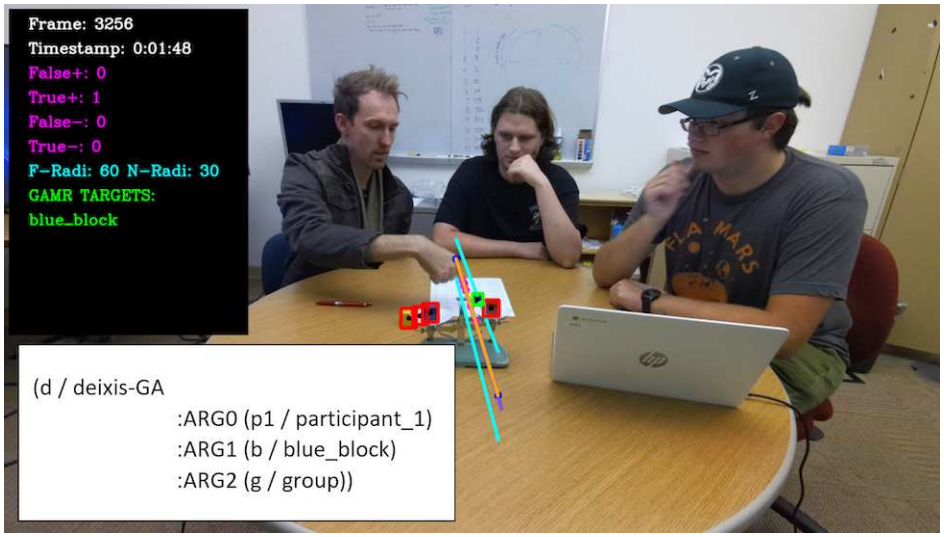


Figure 5.8: Group 1 deixis GAMR example

Figure 5.9: Group 2 deixis GAMR example.

5.2 Experiments and Evaluation

To test the feasibility of the point based target selection, my experimental protocol involved assessing the values of near and far frustum radii that provided the best possible and most consistent object selection across multiple groups. Relevant video frames that were tested against included those which were annotated with GAMR type `deixis-GA`, and had been annotated as containing a point gesture, or the gesture recognizer detected one. From there I assessed which objects

intersected the pointing frustum. Per-frame recall, precision, and F1 were calculated against the `ARG1` of the GAMR annotation. This process was repeated for each relevant frame while keeping a running average for each metric across the entire video. In order to determine the number of correct inferences and type I or type II errors based on selected blocks, I created the following guidelines using the GAMR annotations:

- If `ARG1` is a single block, if the selected block matches the annotation, it is considered a true positive. If selected blocks do not match the annotation, they are considered false positives.
- If `ARG1` is a combination of blocks, such as when the GAMR annotations did not specify a single block as the denoted target, but rather a set, each selected block that matches a block in `ARG1` is considered a true positive. Selected blocks not contained in `ARG1` are considered false positives.
- If a block included in ARG1 is not selected, it is considered a false negative.

The subset of groups I evaluated against included groups 1, 2, 4, and 5 of the WTD (see Section. 2.1).

Table 5.1 shows the average F1, recall and precision across all 4 videos for different combinations of radii, assessed against both the human-annotated pointing frames and those retrieved by the automated gesture detection (see Section. 4.1). It is worth noting the variability in F1 scores; in many cases the standard deviation is almost the same as the average. This indicates the challenge in selecting a single set of frustum radii for the most effective target selection. Additionally, the F1 scores over the the human-annotated frames and the automatically selected frames are generally very similar, showing that our automated pipeline's frame selection achieves similar results when compared to human annotators.

The relatively high standard deviations shown in Table 5.1 indicate the variation present across groups in the dataset. I also present group-wise results showing the average metrics across all frames vs. the radius sizes. This provides additional granularity in determining the most effective

Table 5.1: Average target detection F1 for human annotated and automatically detected frames from groups 1, 2, 4, and 5.

Near	Far	μ Human F1	σ Human F1	μ Auto F1	σ Auto F1
20	50	0.187	0.170	0.185	0.192
30	60	0.213	0.175	0.199	0.191
40	70	0.282	0.254	0.275	0.278
50	80	0.349	0.235	0.338	0.274
60	90	0.401	0.216	0.384	0.267
70	100	0.417	0.207	0.404	0.260
80	110	0.420	0.210	0.404	0.261
90	120	0.418	0.206	0.400	0.261
100	130	0.416	0.195	0.396	0.253

radius combination on a per group basis, as opposed to selecting and testing radius combinations one at a time.

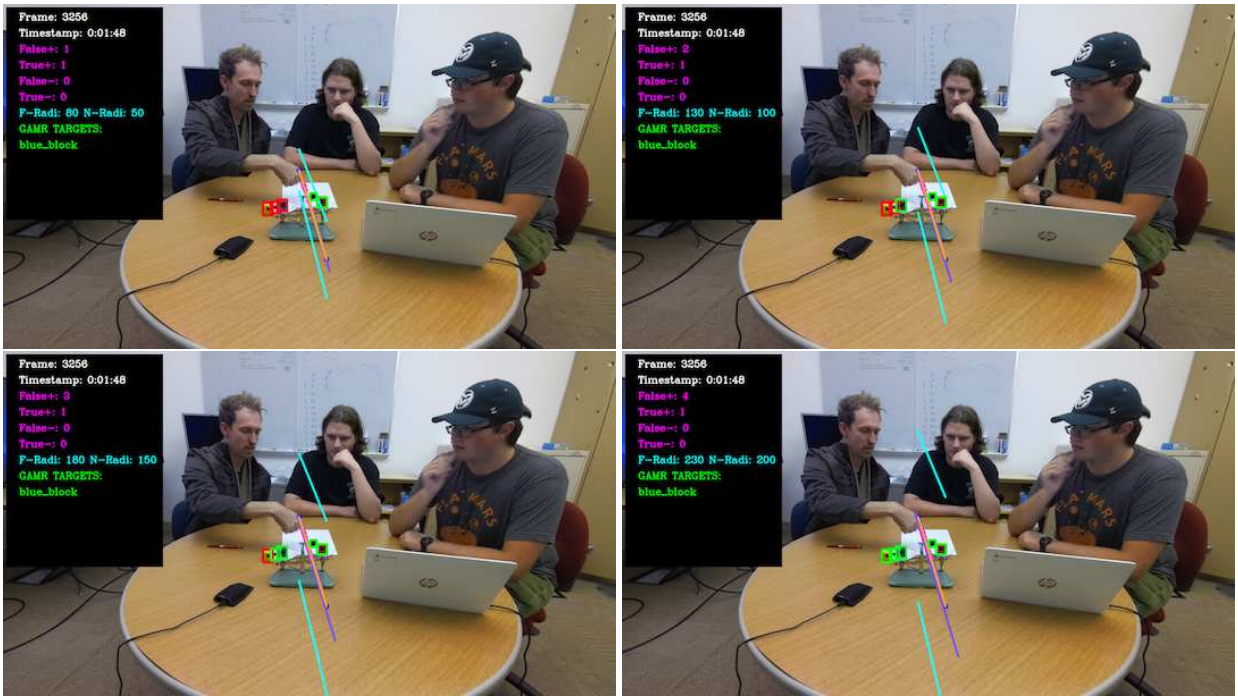


Figure 5.10: Incremental radius step example, Group 1.

Figures 5.10 and 5.11 show examples of the incremental radius steps, and show how as the size of the pointing frustum grows, the detected targets change. Blocks outlined in green indicate those

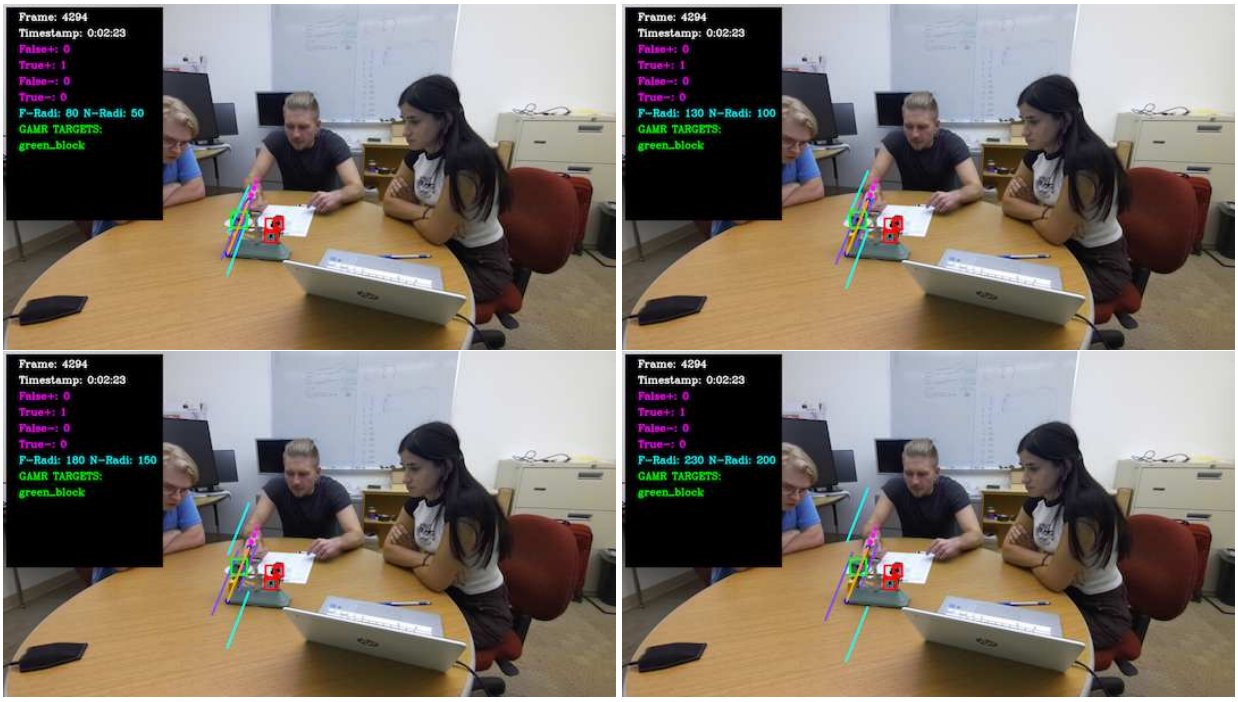


Figure 5.11: Incremental radius step Example, Group 2.

selected as targets of pointing. Red indicates those not selected. Note that in Figure 5.11, as the radius grows the green block is the only one ever selected. This is because the remaining blocks are resting on the paper behind the scale, and are therefore behind the origin of the pointing vector and the frustum's near plane.

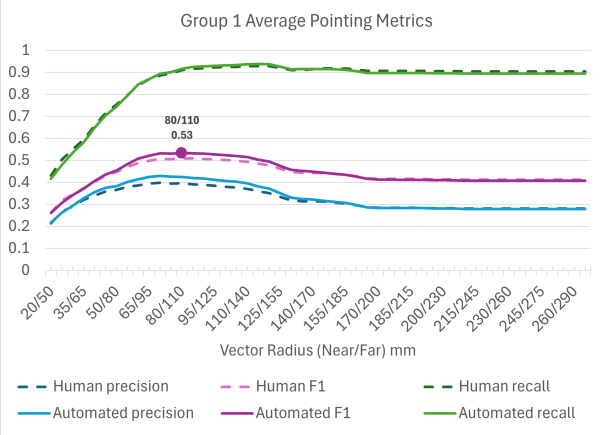
Figures 5.12a–5.12d show experimental results across a range of different near and far radius combinations for each group, averaged across all relevant frames of the video. I compare metrics over those frames annotated by humans (ground truth), and those where pointing gestures were detected by the automated detection pipeline (therefore comprising an end-to-end system with gesture detection and target selection in a single step, see Chapter. 4). Maximum F1 score is indicated with the purple dot. The maximum F1 scores vary anywhere between 0.33 (Group 4) and 0.69 (Group 2). This range is likely due to variability in accuracy and style of pointing as they are used by each participant/group. In addition to human inaccuracy, pointing in such a small space is likely to return more than one object as the radii grow. Because of this, as the frustum size

increases there is potential to return more false positive targets. This is reflected in the increase in recall as the radii grow, but the eventual decline in precision and F1.

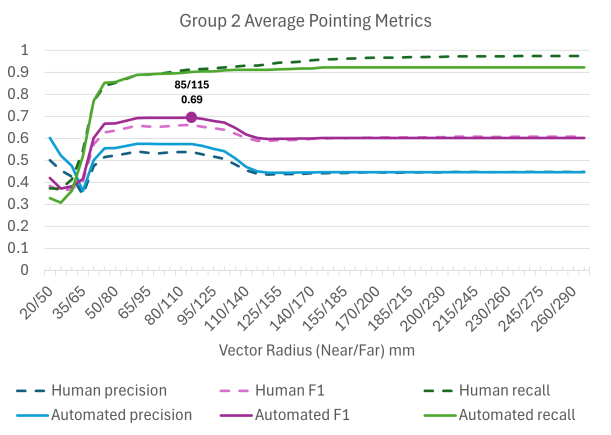
In most cases, except Group 4, using the end-to-end system, where frames were selected by the automated gesture pipeline, outperformed detection over human annotations. In Group 5 particularly, the gesture pipeline frames eventually overtook the human annotated frames by about 0.1 F1 overall and thereafter remained consistent. I hypothesize this may be because the automatically selected frames are ones that the static classification model recognizes as a point with the index finger. In Group 5, sometimes participants would point with pens, or would gesture at the blocks using their entire hand. Overcoming this limitation is another potential area for future work. In other groups the accuracy statistic of the human annotated frames more closely matches the automated frames, indicating that the participants were more likely to point using the index finger (as expected).

The “best" near and far radii values for each group (where the F1 score was the highest) ranges anywhere from 70/100 - 85/115, with an outlier at 185/215 (near/far respectively.) These values are highly dependent on the length of the pointing vector, distance between participants, the size of the work space and the of the objects being detected. That said, in order to utilize these results to determine the “best" radii values for other scenarios linear interpolation could be done using the distance and size values for different scenes and tasks. Outlier values, likely due to differences in pointing style as mentioned previously, may skew this calculation and could likely be ignored. To overcome possible errors in future calculations the experiments are setup in a way where they could be re-run on other tasks further verify this hypothesis.

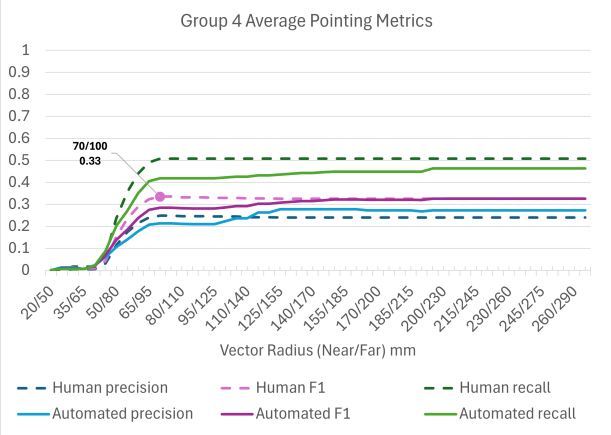
These experiments show the feasibility of robust gesture identification and target selection. These breakthroughs provide a vital link between users and computer vision algorithms that, for example, may meaningfully and efficiently capitalize on users for real time context identification from a scene. Further, the efficiency of the methodology provides a direct platform for real-time human-AI interaction. Future work will need to incorporate features from additional modalities in such a way that allows more accurate target selection, and context extraction for this and other CPS



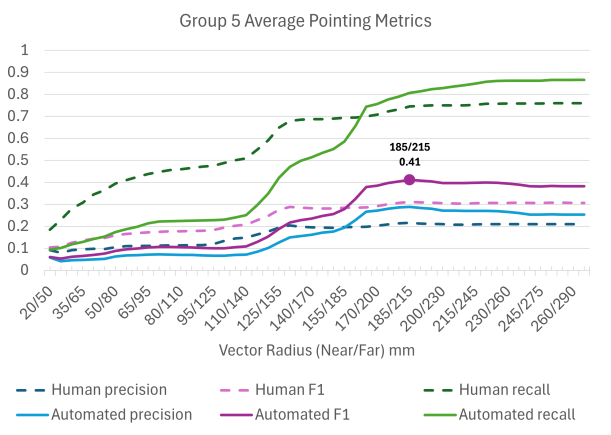
(a) Group 1 target selection statistics.



(b) Group 2 target selection statistics.



(c) Group 4 target selection statistics.



(d) Group 5 target selection statistics.

Figure 5.12: Average precision, recall, and F1 for 4 test groups, averaged over all frames. Solid lines indicate where frame selection was performed using the robust gesture detection pipeline [40]. Dashed lines indicate where frame selection was performed by human annotation.

tasks; however, the addition of additional modalities should balance performance improvements with real-time interaction constraints.

Chapter 6

Applications and Future Work

6.1 Applications

The ability to detect and understand gestures is vital piece for the development of an AI agent whose goal is to interpret and understand small group communication; however, applications for this work extend across a vast amount of additional contexts where gesture is utilized in conjunction with other modalities such as speech, pose and physical space.

Figure 6.1 shows an example of a single interaction in the Weights Task. In this situation, there are many inferences an AI agent could draw about the current state of the task. For instance, P1 and P3 are speaking and gesturing. P3 says "one of these," while performing what appears to be a *grasp* gesture (cf. [40]). With hand and object detection to localize the blocks in the working space, an agent could infer that the group is speaking about the red or blue blocks. This could be used further determine if the overall statement made by P3 is true, thus helping the agent understand the current progress toward successful task completion. P2 is not speaking, gesturing or interacting with the blocks: however, the general direction of their gaze and forward-leaning pose indicate they remain engaged in the task. P1's statement "but I think it's still 20," combined with certain cadence or prosodic patterns could signal P1's confusion or hesitation about the group's current trajectory (cf. [4]). Leveraging the combined modalities, such an agent would be able to maintain a relatively detailed model of what is currently happening in the scene, from specific action-level occurrences to the general level of contribution of each participant.

Additionally, there are many different use cases and design considerations for multimodal systems not only interpret a scene but also return feedback based on small group communication, such considerations are a potential area for continued and deeper research. A well-designed AI agent would collect and interpret enough context from a situation to aid in the problem solving process. For example, it could point out information the group members may not have considered,

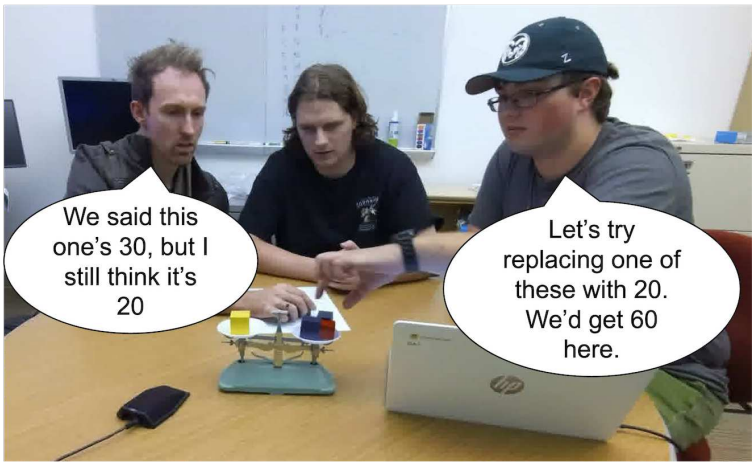


Figure 6.1: Still of Group 1 from the Weights Task. Potential communicative modalities in the scene include but are not limited to, *speech*, *gesture*, *pose*, *action*, *gaze*, and *acoustics*.

realign priorities based on team needs, and incorporate collaborator knowledge on the fly through a representation of objectives, subgoaling, changing plans, uncertainty, etc. An agent would able to interpret the group's general understanding as they work to discover said outcome of a task, based on how they communicate and the CPS skills they display, aid the participants by helping them reach the correct conclusion organically, and could also learn based on participants' task behavior, further tuning its feedback to the group to support their learning in an optimal way. A similar feedback cycle could be applied to any collaborative task, thus empowering teams to think and reason better in a wide range of different tasks and circumstances.

An agent might also take a more direct interactive approach to aiding small groups. One way would be to model the shared and individual beliefs of group members. This would enable it to raise questions about unspoken or unresolved conflicts or intervene in cases of groupthink for which there is no evidence. This is an important capability to prevent groups from making critical errors. As the agent gathers data on communication style and the beliefs of individuals, it would directly intervene and steer the conversation to correct misapprehensions or encourage more productive solutions to relevant subgoals. In a CPS task, this might occur when one or more individuals believe a statement that is incorrect. The agent could step in and ask participants to follow a different line of thought. It could then analyze both if the correct conclusion was drawn from the

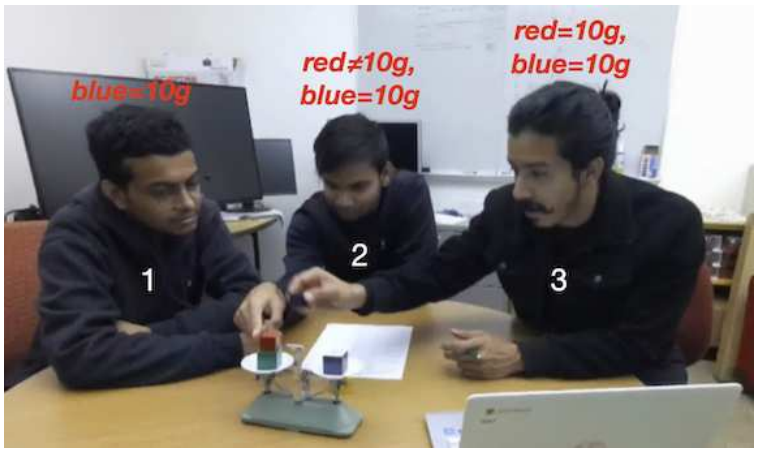


Figure 6.2: Example of a CPS task. A group is working to discover the weight of the blue block, based on inferences about the red block. Red text over each participant shows beliefs each apparently holds at this point, based on their prior utterances and actions.

action, and which CPS skills were displayed during the subdialogue. This interaction style could also be applied to a variety of CPS tasks providing a more direct teaching style for team support.

Figure 6.2 shows a still of a group preforming a CPS task in which they are working to determine the weight of the blue block, based on previous inferences about the red blocks. In this scene multiple different modalities are extracted by an agent to determine the validity of the group's thought process, including but not limited to, what participants are saying, which blocks they are pointing at or grasping, and the location of the blocks relative to the scale. For instance, the whole group believes the blue block weighs 10g, but P2 and P3 disagree about the red block. P3 *acts* based on his belief but P2 makes an inference based on his (expressed in speech).

Both of the aforementioned agent styles would be valid ways to help the group succeed. Leveraging the mentioned example, in a more organic approach the agent would continually collect and process dialogue moves and analyze how the group is working towards the intended goal; for example, by performing inferences over recognized gestures and actions to understand what blocks are being interacted with, object recognition of their locations on the scale, and linguistic understanding of the group's statements to each other, the agent could verify that the current thought process is generally directed toward discovering the correct pattern even if there are specific inaccuracies at the current time. Using a more direct approach, the agent could reason over the inferences of

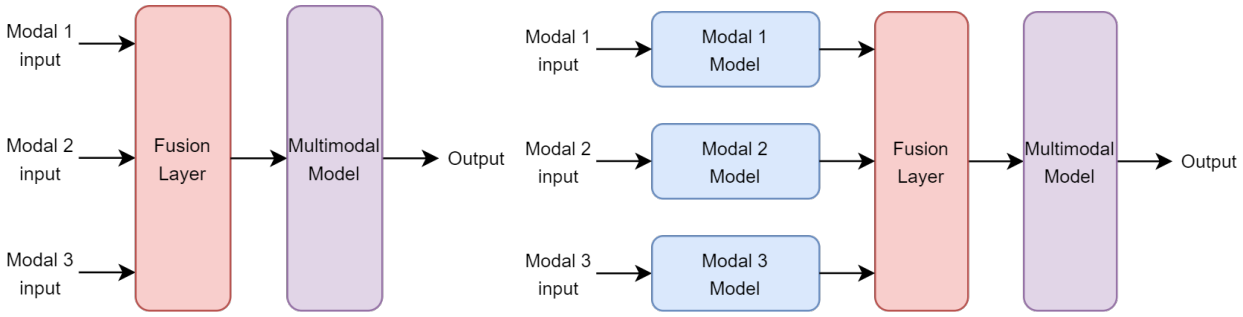
each individual up to the current point in the task and intervene with corrective measures to help drive valid inferences toward the discovery of the overall pattern.

6.2 Future Work

6.2.1 Multimodal Fusion

After identifying and extracting key features of interest, they need to be brought together in a meaningful way so they can be used by an agent. A key foundational challenge of signal fusion is one of aligning the different modal channels, and is thus an important area for future research. For instance, a non-linguistic feature such as gesture, could align with more than one utterance. One potential method to handle feature mapping to utterances involves linking a statement to the feature it overlaps with the most. This causes issues if an utterance spans multiple gestures or actions, and those gestures in question span multiple utterances. To mitigate some of these issues design decisions need to made concerning various fusion methods to bring features together in the most effective way. Design choices in fusion algorithms will primarily revolve around the step at which the fusion of the different feature types takes place, of which there are 3 primary classes: *early fusion*, *late fusion*, or *hybrid fusion* [19].

In *early fusion*, the data is fused at the start of the learning algorithm, such as through concatenation, then processed as a single input. This may lead to imbalance in feature contribution to the final output. *Late fusion* trains on each modality separately in unimodal submodules, and then merges those outputs. This method handles imbalance in feature input size, as the submodules' output sizes are controllable and specifiable. *Hybrid fusion* mixes the previous two where some modalities are trained separately, but if two or more modalities have a certain connection (e.g., overlaps between gesture and action), or if they have the same format and sizes, they may be handled together. Figures 6.3a and 6.3b show high-level schematic diagrams of early and late fusion, respectively. Hybrid fusion combines the two, in that the some modalities may be processed through individual submodules while others are input directly to the fusion layer.



(a) Early fusion schematic diagram.

(b) Late fusion schematic diagram.

Figure 6.3: Diagrams that outline the processes involved in early and late fusion methods

Chapter 7

Conclusion

Small groups engaged in CPS are likely to engage in various forms of multimodal communication, often simultaneously. Potential modalities of interest might include speech, gestures, pose, and interaction with objects in physical space. In order to design an AI agent with the ability track and understand how a group communicates effectively, the ability to extract each communicative feature is vital. However, it's important to note that while each feature of interest is important and provides unique information for use in the overall system, on their own none of the features of interest provide enough information to fully understand what is happening in a scene.

Design considerations concerning the exact methods that will bring features together, in addition to how the participants will interact with the agent remain to be determined and is a potential area for future work (see Chapter 6). A complete system might integrate implementations of 6DOF object recognition, action detection, expression recognition, pose estimation, and speech reconfiguration, in addition to gesture detection [42]. The majority of my research has been based in effective, robust and meaningful gesture detection with the intention of leveraging extracted information for use with other modalities in a future more complete multimodal system.

As an initial, general solution for complex gesture detection, I developed a pipeline for key frame selection based in the foundational work on the semantics of gesture used to model the behavior of humans in multimodal communication, namely the *hold* and *stroke* phases of a gesture [40]. This pipeline aids in the semi-automatic annotation of a range of various gestures, and proved to be a promising solution for extracting key frames of many various gestures that could in turn be used in the overall multimodal system.

One gesture of interest that could be extracted from a video to add meaningful context to a scene are deictic gestures, which are common in small group communication as a means to indicate objects and referents in real time. The ability to both identify deictic gestures, such as pointing, and to identify their denotata in context is a critical capability in interpreting multimodal commu-

nicative acts in situated physical shared tasks. Leveraging my previously-developed pipeline to help automatically detect semantically significant movement for a given gesture, and save off the "key frames" [40], I was able to create a "pointing frustum" based on that information for meaningful target detection [41]. The combination of the automated pointing detection and the "pointing frustum" showed promise as a means to detect the intended target of a participant. However, the experiments also show that gesture alone is not enough to extract complete context from a scene, and additional features are likely needed for an agent to determine the state of any given task. Regarding the deictic gestures, combining this data with the speech feature might add the necessary context to more accurately identify a target. It is worth noting that, the detection pipeline preformed similarly, and in some situations better than, the human annotated ground truth annotation as a reliable, efficient, and robust way of performing object selection via deixis in challenging, real-world data.

Bibliography

[1] Andrews-Todd, J., Forsyth, C.M.: Exploring social and cognitive dimensions of collaborative problem solving in an open online simulation-based task. Computers in Human Behavior **104**, 105759 (Mar 2020). https://doi.org/10.1016/j.chb.2018.10.025

[2] Arnheim, R.: Hand and Mind: What Gestures Reveal about Thought by David McNeill. Leonardo **27**(4), 358–358 (1994)

[3] Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N.: Abstract meaning representation for sembanking. In: Proceedings of the 7th linguistic annotation workshop and interoperability with discourse. pp. 178–186 (2013)

[4] Bradford, M., Khebour, I., Blanchard, N., Krishnaswamy, N.: Automatic detection of collaborative states in small groups using multimodal features. In: Proceedings of the 24th International Conference on Artificial Intelligence in Education (2023)

[5] Breiman, L.: Bagging predictors. Machine learning **24**, 123–140 (1996)

[6] Brutti, R., Donatelli, L., Lai, K., Pustejovsky, J.: Abstract meaning representation for gesture pp. 1576–1583 (Jun 2022), https://aclanthology.org/2022.lrec-1.169

[7] Bugarin, C.A.Q., Lopez, J.M.M., Pineda, S.G.M., Sambrano, M.F.C., Loresco, P.J.M.: Machine vision-based fall detection system using mediapipe pose with iot monitoring and alarm pp. 269–274 (2022). https://doi.org/10.1109/R10-HTC54060.2022.9929527

[8] Castillon, I., Venkatesha, V., VanderHoeven, H., Bradford, M., Krishnaswamy, N., Blanchard, N.: Multimodal Features for Group Dynamic-Aware Agents. In: Interdisciplinary Approaches to Getting AI Experts and Education Stakeholders Talking Workshop at AIEd. International AIEd Society (2022)

[9] Chejara, P., Prieto, L.P., Rodriguez-Triana, M.J., Kasepalu, R., Ruiz-Calleja, A., Shankar, S.K.: How to Build More Generalizable Models for Collaboration Quality? Lessons Learned from Exploring Multi-Context Audio-Log Datasets using Multimodal Learning Analytics. In: LAK2023. pp. 111–121. Association for Computing Machinery, New York, NY, USA (Mar 2023). https://doi.org/10.1145/3576050.3576144

[10] Cunico, F., Carletti, M., Cristani, M., Masci, F., Conigliaro, D.: 6d pose estimation for industrial applications pp. 374–384 (Sep 2019). https://doi.org/10.1007/978-3-030-30754-7_37

[11] D'Mello, S., Graesser, A.: Dynamics of affective states during complex learning. Learning and Instruction **22**(2), 145–157 (2012)

[12] Gibson, J.J.: The theory of affordances. Hilldale, USA **1**(2), 67–82 (1977)

[13] Graesser, A.C., Fiore, S.M., Greiff, S., Andrews-Todd, J., Foltz, P.W., Hesse, F.W.: Advancing the Science of Collaborative Problem Solving. Psychological Science in the Public Interest **19**(2), 59–92 (Nov 2018). https://doi.org/10.1177/1529100618808244

[14] Herbort, O., Krause, L.M.: The Efficiency of Augmented Pointing with and Without Speech in a Collaborative Virtual Environment. In: Duffy, V.G. (ed.) Digital Human Modeling and Applications in Health, Safety, Ergonomics and Risk Management. pp. 510–524. Lecture Notes in Computer Science, Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-35741-1_37

[15] Kandoi, C., Jung, C., Mannan, S., VanderHoeven, H., Meisman, Q., Krishnaswamy, N., Blanchard, N.: Intentional microgesture recognition for extended human-computer interaction. In: Human-Computer Interaction. Springer (2023)

[16] Kendon, A.: Gesture: Visible action as utterance. Cambridge University Press (2004)

[17] Kendon, A., et al.: Gesticulation and speech: Two aspects of the process of utterance. The relationship of verbal and nonverbal communication **25**(1980), 207–227 (1980)

[18] Khebour, I., Brutti, R., Dey, I., Dickler, R., Sikes, K., Lai, K., Bradford, M., Cates, B., Hansen, P., Jung, C., et al.: When text and speech are not enough: A multimodal dataset of collaboration in a situated task. Journal of open humanities data **10**(1) (2024)

[19] Khebour, I., Lai, K., Bradford, M., Zhu, Y., Brutti, R., Tam, C., Tu, J., Ibarra, B., Blanchard, N., Krishnaswamy, N., Pustejovsky, J.: Common Ground Tracking in Multimodal Dialogue. In: Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING). ACL (2024)

[20] Kita, S.: Pointing: A foundational building block of human communication. Pointing: Where language, culture, and cognition meet pp. 1–8 (2003)

[21] Kong, A.P.H., Law, S.P., Kwan, C.C.Y., Lai, C., Lam, V.: A coding system with independent annotations of gesture forms and functions during verbal communication: Development of a database of speech and gesture (dosage). Journal of nonverbal behavior **39**, 93–111 (2015)

[22] Kranstedt, A., Lücking, A., Pfeiffer, T., Rieser, H., Wachsmuth, I.: Deixis: How to determine demonstrated objects using a pointing cone. In: Gesture in Human-Computer Interaction and Simulation: 6th International Gesture Workshop, GW 2005, Berder Island, France, May 18-20, 2005, Revised Selected Papers 6. pp. 300–311. Springer (2006)

[23] Krishnaswamy, N., Alalyani, N.: Embodied multimodal agents to bridge the understanding gap. In: Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing. pp. 41–46 (2021)

[24] Krishnaswamy, N., Narayana, P., Bangar, R., Rim, K., Patil, D., McNeely-White, D., Ruiz, J., Draper, B., Beveridge, R., Pustejovsky, J.: Diana's world: A situated multimodal interactive agent. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 13618–13619 (2020)

[25] Krishnaswamy, N., Narayana, P., Wang, I., Rim, K., Bangar, R., Patil, D., Mulay, G., Beveridge, R., Ruiz, J., Draper, B., Pustejovsky, J.: Communicating and acting: Understanding

gesture in simulation semantics. In: IWCS 2017—12th International Conference on Computational Semantics—Short papers (2017)

[26] Lascarides, A., Stone, M.: A formal semantic analysis of gesture. Journal of Semantics **26**(4), 393–449 (2009)

[27] Li, J., Jin, K., Zhou, D., Kubota, N., Ju, Z.: Attention mechanism-based CNN for facial expression recognition. Neurocomputing **411**, 340–350 (Oct 2020). https://doi.org/10.1016/j.neucom.2020.06.014

[28] Li, S., Deng, W.: Deep Facial Expression Recognition: A Survey. IEEE Transactions on Affective Computing **13**(3), 1195–1215 (Jul 2022). https://doi.org/10.1109/TAFFC.2020.2981446

[29] Lücking, A., Bergmann, K., Hahn, F., Kopp, S., Rieser, H.: The bielefeld speech and gesture alignment corpus (saga). In: LREC 2010 workshop: Multimodal corpora–advances in capturing, coding and analyzing multimodality (2010)

[30] Mather, S.M.: Ethnographic research on the use of visually based regulators for teachers and interpreters. Attitudes, innuendo, and regulators pp. 136–161 (2005)

[31] McNeill, D.: Hand and mind. Advances in Visual Semiotics **351** (1992)

[32] McNeill, D.: Gesture and thought. In: Gesture and Thought. University of Chicago press (2008)

[33] Narayana, P., Beveridge, R., Draper, B.A.: Gesture recognition: Focus on the hands. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5235–5244 (2018)

[34] Pustejovsky, J., Krishnaswamy, N., Do, T.: Object embodiment in a multimodal simulation. In: AAAI Spring Symposium: Interactive Multisensory Object Perception for Embodied Agents (2017)

[35] Singh, A.K., Kumbhare, V.A., Arthi, K.: Real-Time Human Pose Detection and Recognition Using MediaPipe pp. 145–154 (2022). https://doi.org/10.1007/978-981-16-7088-6_12

[36] van der Sluis, I., Krahmer, E.: The influence of target size and distance on the production of speech and gesture in multimodal referring expressions. In: Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP'04) (2004)

[37] Stewart, A.E.B., Keirn, Z., D'Mello, S.K.: Multimodal modeling of collaborative problem-solving facets in triads. User Modeling and User-Adapted Interaction **31**(4), 713–751 (Sep 2021). https://doi.org/10.1007/s11257-021-09290-y

[38] Sun, C., Shute, V.J., Stewart, A., Yonehiro, J., Duran, N., D'Mello, S.: Towards a generalized competency model of collaborative problem solving. Computers & Education **143**, 103672 (2020), https://www.sciencedirect.com/science/article/pii/S0360131519302258

[39] Sun, C., Shute, V.J., Stewart, A.E., Beck-White, Q., Reinhardt, C.R., Zhou, G., Duran, N., D'Mello, S.K.: The relationship between collaborative problem solving behaviors and solution outcomes in a game-based learning environment. Computers in Human Behavior **128**, 107120 (2022)

[40] VanderHoeven, H., Blanchard, N., Krishnaswamy, N.: Robust motion recognition using gesture phase annotation pp. 592–608 (Jul 2023). https://doi.org/10.1007/978-3-031-35741-1_42

[41] VanderHoeven, H., Blanchard, N., Krishnaswamy, N.: Point target detection for multimodal communication. In: Digital Human Modeling and Applications in Health, Safety, Ergonomics and Risk Management. Springer (2024)

[42] VanderHoeven, H., Bradford, M., Jung, C., Khebour, I., Lai, K., Pustejovsky, J., Krishnaswamy, N., Blanchard, N.: Multimodal design for interactive collaborative problem-solving support. In: Human-Computer Interaction. Springer (2024)

[43] Wolf, K., Naumann, A., Rohs, M., Müller, J.: A taxonomy of microinteractions: Defining microgestures based on ergonomic and scenario-dependent requirements. In: 13th International Conference on Human-Computer Interaction (INTERACT). pp. 559–575. No. Part I, Springer (2011)

[44] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.L., Grundmann, M.: Mediapipe hands: On-device real-time hand tracking. arXiv preprint arXiv:2006.10214 (2020)