**Background**

The team inherited the work of a team pursuing a spot in the AI XPrize competition. In this process, the team envisioned an end goal of a system where a human laborer commanded a responsive robot to move blocks. The idea was to empower the human worker, whose ingenuity could be unrestricted by physical fatigue by allowing a robot to perform the actual actions. While the team was not able to produce a prototype that accomplish these goals, they did provide many artifacts that will aid us in pursuing these objectives. Two of these artifacts will be explained in detail here, as they serve as restrictions on future project planning.

The first is the chosen architecture for the overarching system. Last year’s team foresaw the need to combine inputs from a variety of different sources, despite primarily working with text input. To accommodate this need for input integration, the team chose UIMA. This architecture allows for new inputs to be added to processing through an arbitrary number of processing components, which each pass structured, analyzed information to a central processor.

Typically, this architecture suffers from a lack of language flexibility, as existing UIMA support is not provided in many languages in which components may be implemented. To remedy this issue, last year’s team created a HTTP-based interface to allow for information to be sent from a component to the central processor in a language agnostic mode. In doing so, the team ensured that the architecture would be flexible enough to allow for any form of component needed. Actually using this architecture for an integrated system would require massive amounts of work once each component was in place, but the architectural decision still laid the groundwork for future work.

The second major inherited decision was related to the most fleshed out UIMA component, that being the text processing. Last year’s team created a “Blocks World” game in which users would provide text commands and emulate the actions a robot would take to complete a sequence of actions. This game was entirely 2D, so much of the location recognition and task completion will not have much utility for a practical application. However, the text processing does have utility. While an ideal setup would utilize speech, there are external libraries that handle the conversion from voice to text. The result is that the network trained for text processing can likely be applied with modifications as a component of an integrated system, albeit perhaps with transfer learning first performed to obtain a more pertinent network.

**Our Overarching Research Goals**

We envision the final setup consisting of a fixed sensor overlooking a table-sized interaction space. This sensor will aid in recognizing a set of predefined gestures, from which we can continue processing accordingly, whether this is to identify the object being pointed to, the point to move to, or some other domain-relevant task. Ideally, this gesture set can begin small and expand as needed to make collaboration simpler, building on previous research in this field to expedite the actual work needed to implement gesture recognition.

On that note, we outlined a set of intermediary research goals that could serve as targets to move towards as we explore the technologies and techniques involved in this domain.

1. Identify requirements for ideal end system.
   1. Ex: Other gestures of need.
2. Abstract a single pointing gesture into a vector.
3. Process a single pointing gesture to identify the object pointed to.
4. Combine with language processing to fully disambiguate blocks.
5. Integrate system with feedback into the existing UIMA pipeline.

These objectives ramp up in difficulty quickly and may possibly expand beyond the work that can be completed this year. We do note that the later objectives are closer to the domain of image processing than machine learning and thus may allow us to circumvent the associated volumes of data needed for training. Regardless, these objectives are entirely arbitrary and may be adjusted as stakeholders identify other, more promising directions to take our research. In particular, if possible grants necessitate the pursuit of other work then doing so seems more fruitful than pursuing the above outline. In either case, some form of gesture processing must be able to address the above objectives in some fashion, making them ideal targets to segment our work moving forward.

For each of the above phases, we will identify a research hypothesis (if relevant), a description of the work that needs completion, and provide requirements that are specific to that phase of research. Note that these “prototype-specific requirements” are in addition to, not a replacement of, the requirements identified for previous prototypes. Ideally, each prototype will build on the previous ones to prove the feasibility of solutions to increasingly complex problems in our domain.

**Constraints after Initial Research**

The above research goals emerged after an initial round of research into gesture recognition. In doing this research, we provide our client (Dr. Wollowski) with information that caused him to impose a few restrictions onto our research. These restrictions are enumerated below and will be permitted into user stories as assumed technical details. Other technical details imply a particular implementation approach and thus will be left out of user stories. This list will grow and evolve as additional restrictions are imposed.

* The Kinect sensor will be explored explicitly.
* Ignore gesture identification and instead assume the person is always pointing. Will re-evaluate once pointing processing is complete.
* Attempt entirely algorithmic approach before attempting any machine learning.
* Assume the background is white.
* Assume the blocks are of a different color than the background.
  + (These assumptions about the colors makes block detection a substantially simpler problem).

**Phase 1: Identify requirements for ideal end system**

**Description**

When reflecting on last year’s work for object placement, all stakeholders indicated that the team was overly optimistic about how much of their work would transfer to the 3D space that the final system would occupy. After initial discussions, our team determined that the previous team did not properly consider and/or adhere to the desired attributes of the end system. Instead, the team sought to implement a system (i.e. their “Blocks World” prototype) and see how much could be transferred to the final problem domain. Though much of their work transfers to varying degrees, it is especially important that we have a clear idea of important quality attributes of the ideal end setup to avoid completing research goals that do not actually contribute to the overarching system.

As such, below are a list of overarching requirements that will apply to the end system. Intermediate requirements specific to particular research goals will be defined as particular goals are tackled. These goals serve as guidelines that should restrict, but not necessarily define, the intermediate research goals. This list is non-static and serves as an evolving state of our stakeholders’ desires.

**Produced Requirements in Separate Document**

**Phase 2: Abstract a pointing gesture into a vector**

**Research Hypothesis**

A single vector provides sufficient information to completely capture a pointing gesture. Converting a pointing gesture to a vector can be done algorithmically.

**Description**

The next component to implement is gesture recognition. In particular, it is common for a user to say phrases like “move *that* block”, where “that” is a clearly ambiguous quantifier. To understand exactly what “that block” is, we need to have some visual component. This visual component is understandably complex, so the first goal is simply to recognize the clearest gesture in pointing. It is easy to envision a user augmenting a phrase like “that block” by pointing at the block of interest. This pointing gesture can be further reduced into a simple vector, which provides the positional context behind a phrase like “that block”.

Given this context, our first foray into gesture recognition is restricted to a single pointing gesture. However, it is possible that additional gestures will be needed in the end system. Thus, while we cannot articulate this restriction easily into a clear requirement, we will impose the additional restriction that our implementation cannot be one that precludes the possibility for additional gestures in the future. That is, the architecture that guides our implementation must accommodate the possibility of additional gestures and thus be extensible/modular in this regard.

**Prototype-Specific Requirements**

1. The system will only require the user’s pointing arm be in view
   1. As opposed to needing their whole body in view.
2. The user is assumed to always be pointing.
   1. Non-pointing gestures will not have meaningful output.
3. The system will display the user’s hand.
4. The system will display the computed vector.
   1. This vector is a mathematical abstraction of the user’s pointing gesture.
5. The system will capture the entire shared collaboration space.
   1. As opposed to being focused solely on the user.
6. The system will be tolerant to slight changes in lighting.
7. The system will be tolerant to different hand angles relative to the input sensor.
8. The system shall be capable of growing to recognize other gestures.

**Phase 3: Process a single pointing gesture to identify the object pointed to.**

**Research Hypothesis**

Ray tracing a vector abstraction for a pointing gesture is a sufficient measure to find the closest object in a space.

**Description**

Depending on implementation, this objective is not necessarily separate from phase 2. However, for most practical cases, there is a large leap from recognition of a pointing gesture to application of that pointing gesture for spacial awareness. This point of processing is where the system must have some fundamental understanding of the space. That is, there must be some way to associate a gesture with a nearby (not just any) object. For the intent of this project, these objects are blocks in particular, but our approach should not be so intimately tied to the object used that modifications cannot be made to accommodate new objects in the future.

Processing at this stage is intended to apply to disambiguate one of several blocks. Hence, this prototype must also consider how it will detect the blocks in the collaboration space. Recognition would not need to be in the form of a robotic pointing arm (which would imply a greater degree of integration with the mechanical end), but should be a format that is meaningful for future processing.

**Prototype-Specific Requirements**

1. The system’s video input will not center on the collaboration space.
2. The system’s video input will not be centered on the human collaborator.
3. The system will provide a confidence rating which encapsulates the likelihood a block is pointed to.
4. The system will output the closest block if one is above a certain confidence.
5. The system will not output any block if no block is above a certain confidence.

**Phase 4: Combine with language processing to fully disambiguate blocks.**

**Research Hypothesis**

The combination of language and gesture for block detection is better than either in isolation.

Language and gesture components can be trained independently and integrated with minimal retraining.

**Description**

Most of last year’s work centered on language processing, so we opt to tackle gesture processing first before returning to their work. Language provides meaningful descriptors about the target block, including color, relative location, and various such restrictions that can constrain the blocks considered. With gesture, we used location and human pointing alone to disambiguate blocks, which only required a simple abstraction of blocks into locations. However, to leverage language, we need to introduce details to our abstraction that will allow us to filter based on given descriptors. The most notable addition is that of color, but details like size or shape may be relevant distinguishers to consider.

Primarily, using language allows us to provide the human collaborator with the two most natural ways to instruct our system to complete a task. Indeed, integrating a confidence based on language with a confidence based on gesture is of particular interest, as these two systems will likely operate in independent environments. Thus, we want to determine whether this integration can be done without extensive amounts of retraining, thereby “vertically bootstrapping” these systems into one processing unit.

**Prototype-Specific Requirements**

* NOTE: Open to change, will be expanded on and may be modified after learning of pitfalls from gesture-only processing.

1. The system will identify and use block color as a distinguishing factor between blocks.
2. The system will only perform gesture processing after a collaborator’s issues a command.
   1. Hence, image processing is not continuous.
3. The system will only recognize commands that move blocks.
4. The system will not output a block if the collaborator’s instruction is not decipherable.

**Phase 5: Integrate system with feedback into the existing UIMA pipeline.**

**Research Hypothesis**

Using UIMA does not require retraining to combine language and gesture processing components.

A feedback loop can be used to evolve a system’s abstracted understanding of the environment.

**Description**

It is entirely likely that the initial attempt at integration language and gesture will use the networked UIMA architecture produced by last year’s team. As such, this phase may have one of its research goals entirely encompassed by the previous phase. However, the system has minimal use without substantive feedback. Our far-reaching research objectives center on building a shared context between the collaborator and the robot. To build this context, the robot needs to be able to identify information about previous commands. That is, the robot must be able to confirm, at a minimum, whether it’s predictions were successful and how it can clarify ambiguity in a collaborator’s commands.

This feedback must also have a substantial impact on the system, as soliciting this feedback is of no value unless the robot uses that feedback to improve future predictions. For now, we will not focus on a robust form of improvement, but simply acknowledging the success or failure of predictions and recognizing how commands change when ambiguity is clarified provides a good basis for evolving an understanding of the environment. This understanding must be represented in some way, one which allows a developer to justify why the system made a particular prediction.

**Prototype-Specific Requirements**

* NOTE: Open to change, will be expanded on and may be modified after performing initial component integration.

1. The system will prompt the collaborator for another command if no output block is produced.
2. The system will ask the user if its prediction is correct if an output block is produced.
3. The system will log details about its confidence for its prediction.
4. The system will log details about its abstracted representation of the environment after completing an instruction.