Improving Human Computer Interaction in a Classroom Environment using Computer Vision

Joshua Flachsbart, David Franklin and **Kristian Hammond**
Intelligent Information Laboratory
Northwestern University
1890 Maple Avenue
Evanston, IL 60201 USA
{josh, franklin, hammond}@infolab.nwu.edu

ABSTRACT
In this paper we discuss our use of multi-modal input to improve human computer interaction. Specifically we look at the methods used in the Intelligent Classroom to combine multiple input modes, and examine in particular the visual input modes. The Classroom provides context that improves the functioning of the visual input modes. It also determines which visual input modes are needed when. We examine a number of visual input modes to see how they fit into the general scheme, and look at how the Classroom controls their operation.

Keywords
Intelligent Environments, Multi-modal Input, Computer Vision, Context-Based Vision

INTRODUCTION
In most computer systems, successful interaction occurs when the user understands the constraints of the input devices, and has built up an expertise in using them. These rules are learned once and, depending on how well the system maintains its method of interaction, used for many different tasks. These interfaces are unfortunately often blind to the world in which they exist. This means that outside of the given constraints the input device is useless. For example, when a speaker gives a talk using computer based slide projectors, a mouse click is generally used to indicate when the next slide is needed. This can be difficult if the user likes to walk around the room, because the constraint for a simple mouse is that it be connected to the computer. More generally, being tied to a single interaction method or input device can be very limiting for a user, depending on the given task. We will refer to the input portion of a given interaction method, or input device as an input mode.

We are currently developing a system that eliminates the user's reliance on a single constraining input mode in two ways. First, we allow the computer to have a number of ways of interacting with the user. If the constraints aren't appropriate for a given task then a different input mode with more appropriate constraints can be used. Second, we allow the computer to be aware of both the environment that the user is in and the state of the user. In our current system the state of the user is simply the task in which the user is engaged. By allowing the system to sense the world and the user, it can know what constraints currently hold. Using this information the computer can pick the best method of interaction for the given state of the world. This system allows the user to interact with the computer in the best way for any given task.

The use of the word interaction is important here. Standard computer input devices place the entire burden of interaction on the users. It is hard to call what normally happens with users interaction; the computers simply react to the commands of the users. By placing some of the burden of interaction on the computers we hope to make the interaction with computers be more like interaction with other people.

In order for this type of interaction to work, the computer must have an understanding of what the user is doing, and what the goals of those tasks might be. In standard systems, the user gives the computer a command and the computer executes it. In our system the computer watches the user and figures out what they are trying to do. If there is a step in the plan the user is executing that the computer can do for the user it will execute it for the user.

Our system is implemented in a physical environment, the Intelligent Classroom. The Classroom allows us to test whether our interaction methods work, and whether our method selection scheme works. To provide unique methods of interaction the facility has been equipped with microphones and cameras to provide input to the computer as well as computer controlled audio-visual devices. The classroom can then observe the lecturer and act as if it were under the control of an audio-visual technician, freeing the lecturer from many of the technical details of giving a lecture.

We begin this paper by taking a quick look at the Intelligent Classroom and the tasks that it currently performs. We then look at the visual input modes that are currently implemented in the Classroom. The Classroom is a work in progress and still has a number of features that need to be implemented, so we will look at some of the

future directions of the classroom. Finally we look at some related work and make some concluding remarks.

THE INTELLIGENT CLASSROOM

We are constructing the Intelligent Classroom as a research platform for developing new methods of Human Computer Interaction. The main areas of research covered by the classroom include sensing – vision and speech understanding – plan recognition, and plan execution. The goal of the Classroom is to have a system that can interpret the actions of the user, and then act itself to help the user. Currently the Classroom considers the lecturer or speaker to be the only user of the system, so it attempts to aid the lecture's progress. For example, if the user started a slide show and asks for the next slide, the room changes slides for the user.

The Classroom senses its world with cameras and microphones, and affects its world with computer controlled VCRs, projectors, lights and televisions. By controlling these multimedia devices, the Classroom frees the lecturer from worrying about how to run all of the device, and allows the lecture to flow more smoothly. In addition, the Classroom is equipped with a camera which it uses to make a video of reasonable quality of the lecture, for transmission to another site or for later use. This video follows the user, and frames the scene depending on the context of the lecture. For example, if the user is writing on the chalkboard, the camera zooms in on the board so that the person watching can see what the user is writing. The techniques for sensing and effecting the world allow interaction with the Classroom to be more natural, and non-intrusive. A full description of the tasks and goals of the Intelligent Classroom project can be found in [5].

Interaction in the Classroom: A System that Understands Your Needs

Our main goal is to have a system that can understand your needs and fulfil them. In order to do that, it needs to know something about the tasks that you are doing, and what the goals of those tasks are. It therefore also needs to be able to tell when you are doing one task or another.

For example, the Classroom must be able to sense when the user needs the slide show to be started, and that the next thing the user might do is turn on the projector and load the slides for this lecture. Another example might have the Classroom filming the lecture. In order to do this, it must know where the user is as well as whether he is writing on the chalkboard. By reasoning about the actions it sensed and the consequences of those actions, the Classroom can determine the goals of the user.

By using these goals, the Classroom can determine incomplete steps in the plan to attain those goals, and help the user by accomplishing those steps. For a full description of the plan recognition and execution system in the classroom, see [6]. This knowledge of the state of the world also allows the Classroom to reason about the operation of its internal sensing mechanism, and direct it according to the current context.

Because the Classroom has a number of input modes available to it, and they are not all valid all of the time, it needs to be able to reason about the input modes. The Classroom must also have a reasonable model of the world to understand which actions will result in which consequences, and which sensing modes are appropriate at given times.

Tasks for the Intelligent Classroom

The Intelligent Classroom provides a useful environment to test methods of interaction because of the broad range of tasks in which a user may choose to engage, and the number of ways in which the Classroom may respond. The current incarnation of the Classroom has the following set of tasks that it can accomplish:

- Produce a film of the user giving a lecture.
- Run a PowerPoint slide shows for the user.
- Control VCRs for the user.

In giving a lecture, the user may at any point be doing a number of different things. These things will often end in one of the above tasks needing to be done. For example, if the user is giving a slide show, he will need the slides to be changed at some point. The Classroom should be able to recognize when that point is and change the slide for the user. In order for this interaction to be effective, the user must be able to inform the Classroom of what needs to be done. The input mode that the Classroom employs to sense the user depends on the task.

INPUT MODES FOR DIFFERENT TASKS

In order to allow the Classroom to interact naturally with the user, we have developed a number of input modes. The rest of this paper will look at the different modes and see how they work for the different tasks.

Making Decisions about Input Modes

The Classroom must consider a number of factors when deciding which input mode will be the best for a given situation including: ease of use, correctness and efficiency. Some interactions might be correct, but not very efficient. For example, going to a keyboard to switch to the next slide lets the computer know exactly what needs to be done when, but forces the user to move to the computer. On the other hand simply speaking and expecting the Classroom to know when to switch the slides based on the content of the slides is much more efficient, but can be less accurate.

The job of incorporating all of these considerations into the system lies with the input mode designer. In general an input mode will be crafted for a specific or a number of specific tasks. A given input mode generates a specific type of information that is required by one or more tasks. We will now look at a number of different visual input modes currently implemented in the Classroom, and the tasks for which they are designed. We will then look at how the Classroom will reason about them.

VISUAL INPUT MODES

General vision is a hard problem, but many researchers have solved individual cases of vision problems by relying on the constraints specific to that visual task[1,2,7,11]. The Classroom facilitates effective vision in a number of different contexts by using input modes whose constraints

hold in the given situation. In order to achieve this, the constraints of the input modes must be explicit, and the Classroom must be able to modify the input modes.

All of our visual input modes are built in a versatile vision architecture called Gargoyle. Gargoyle was designed with the intention of being used in a dynamic environment like the Classroom or a mobile robot. The Gargoyle system meets the Classroom's visual requirements by allowing the visual routines to be reconfigured at run-time. Visual routines are specific visual tasks that can be used as input modes in the Classroom. Gargoyle allows visual routines to be activated and deactivated as required. Additionally Gargoyle allows the Classroom to switch individual elements of a visual routine, called modules, as required by the changing context. For a full description of Gargoyle see [4].

We currently have two visual tasks implemented in Gargoyle for the Classroom. One is person tracking and the other is icon recognition. Each of these visual tasks is designed for specific problems that the Classroom needs to be able to solve to interact well with the user. We will now look at some of those problems and how they influence the design of the input modes.

TASK REQUIREMENTS

Since our vision system relies on constraints, none of the visual input modes that we use can exist in a vacuum. Each input mode is designed with a specific task in mind. We have three tasks that are currently solved by visual techniques. We now briefly examine each of the tasks and the requirements they place on the input modes.

Plan Recognition

Plan recognition is a crucial task for the Classroom because it allows the Classroom to cooperate with the user. Knowing what the user is doing also provides task constraint which helps the Classroom choose appropriate input modes. The Classroom is able to make some assumptions about the user's goals by observing where he is in the room. For example, if a user is writing on the board, he is in the middle of a "writing on the board task." Recognizing the user's plans requires reasonably accurate tracking information, and hand location is valuable. Along with verbal cues other input modes, like speech recognition, provide helpful information to the plan recognizer.

Filming a Lecture

In order to produce a reasonable quality video of the lecture the Classroom needs a current estimate of not only where the user is in the room, but also what he is doing. For example if the user is showing a slide show the Classroom should put the slide into the video stream. The key for producing a good film is that the input needs to be fast, if not quite as accurate.

Using Control Icons

The Classroom also allows the user to interact with it through control icons. The icons take two forms, ones that the Classroom displays on the projection screen, and ones that the user draws on the whiteboard. In order to tell when the user is touching a control icon the Classroom needs to track the hands of the user quite accurately. Additionally the Classroom needs to be able recognize any control icons that the user draws on the board.

PERSON TRACKING

All of these tasks require the Classroom to know where the user is. This is accomplished by visually tracking the user as he moves around in the front of the facility. The tracking visual task can be accomplished using one of two different input modes. The first utilizes background subtraction. The other input mode for person tracking utilizes a color model of the user and color histogram backprojection. We now examine both of these input modes in detail.

Background Subtraction Input Mode

The background subtraction input mode is implemented in five main parts, which are Gargoyle modules. The first three modules are: *background subtraction*, *light finding*, and *leg finding*. These feed into a *combiner* module, which in turn feeds the *person finding* module. Figure 1 shows the layout of the person tracking pipeline, with the light finding module removed.

The first module uses simple background subtraction to segment the user from the rest of the background. The background subtraction module memorizes the background by creating a histogram of the grayscale values for each point. As each new image comes in, the background is updated, and the pixels of the input that are within some distance of the mode are considered background. The rest of the pixels are sent to the morphological segmenter as candidates for the user's location.

This technique assumes a stationary camera, and only one person in the scene at a time. The room can enforce the stationary camera criterion when it is using this method. In general, only one user will be lecturing at the front of the Classroom, so the second assumption is usually safe. The morphological segmenter determines when this constraint fails. One additional constraint on this module is that the user not stand in one place for too long. This is because it continuously updates the background, and the user will eventually fade into the background. The future work section shows examples of how the Classroom can deal with failures of these constraints.

The other modules that feed the person tracker module mainly clean up the output from the background subtraction module. The light finder is a simple threshold module. This allows the person tracker to ignore foreground from the apparent motion caused by the flicker in the lights in the room. When the lights are not visible in the scene, the Classroom leaves the light finder module out of the pipeline.

The leg finder eliminates foreground caused by the motion of the user's shadow. The leg finder is actually two modules. First, an edge detection module which can see the edges of the user's legs, but not the soft edges caused by the user's shadow on the wall. Most institutional walls are untextured solid colors, meaning that only the user's legs show up in the lower part of the image. The user normally casts shadows on the lower part of the wall, which will

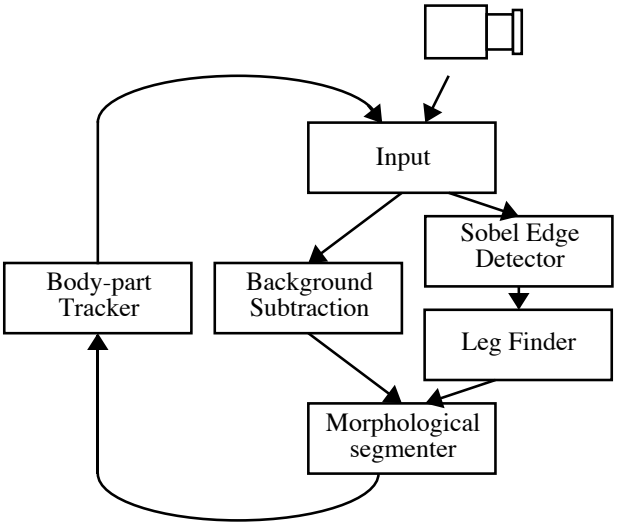

Figure 1: Background Subtraction Pipeline

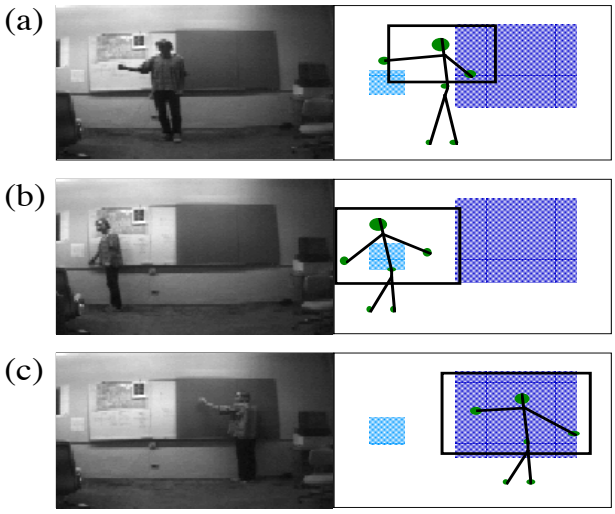

Figure 2: Classroom Representation of the user.

confuse the background subtraction module, but not create any large sharp changes that the edge detector will pick up. Any pixels in the lower part of the scene that are not textured are not considered to be part of the user. This creates a constraint in the background subtraction input mode that the wall be untextured.

The output of these modules are then combined and handed to a morphological segmenter or person finder module. The person finder module provides all of the information about where the user is to the Classroom. The Classroom in turn provides constraints on where the body parts can be and move to back to the input mode. The module does a connected component analysis on the combined image to find the large foreground element, which is considered to be the user. It then examines different parts of this shape to determine where the head, hands and centroid are. An example of the Classroom's representation of the room and the user can be seen in Figure 2. The images on the left are the input from the Classroom's camera. The images on the right are the Classroom's representations for each of the input images. The dark shaded region is where the Classroom thinks that the chalkboard is, and the outlined box is what it thinks the appropriate region to frame for the film is.

The centroid information is very accurate and can be used to film the user with a high degree of reliability. The centroid information is also used by the plan recognizer to reduce the possible number of plans that the user can be executing. For example, if the user is writing on the white-board while lecturing, he must be at the white-board. The hand information is less reliable, especially if the hand is in front of the body, in which case the hand is not sticking out to the side. This representation of the user is, however, sufficient for recognizing a number of gestures. Significantly, it allows us to recognize where the user is pointing or what the user is touching. For example, if the user is giving a slide show, he may opt to touch the screen in a specific location to have the Classroom advance the slide for him. It also allows the Classroom to understand when the user is touching, or drawing a control icon, described in the next section.

Color Histogram Input Mode

The color histogram backprojection input mode is very similar to the background subtraction mode, but it has a number of different constraints. The most important constraint is that the Classroom has built up a current color histogram of the user. This is normally done ahead of time to assure an accurate histogram. It can also be done using the background subtraction technique to segment the person pixels, and then determine the histogram from those pixels. The histogram tracking method also assumes that the histogram of the user will not change.

The module finds the person by comparing the distance of different parts of the image to the given histogram. By deciding at each point if it matches the histogram or not, the module is able to make “person pixel” and “non-person pixel” determinations and produce an output similar to that of the combiner module in the background subtraction mode.

Given that the outputs are so similar, we use the same morphological segmenter and body part tracker as before. In general we tend not to track all the body parts because the color histogram output is not as sharp or accurate on deciding person and non-person pixels as the background subtraction mode. Because of this the color histogram input mode only tracks the centroid and the head. This means that it is not appropriate for tasks which require knowing where the hands are. On the other hand it is faster because the morphological operations are cheaper, and therefore more appropriate for tasks which require speed. The color tracking input mode also has many fewer constraints than the background subtraction mode and thus can be helpful when that mode fails.

ICON RECOGNITION FOR VIDEOS AND SLIDES

The icon recognition visual routine recognizes icons that the user may draw on the white-board. It does this by

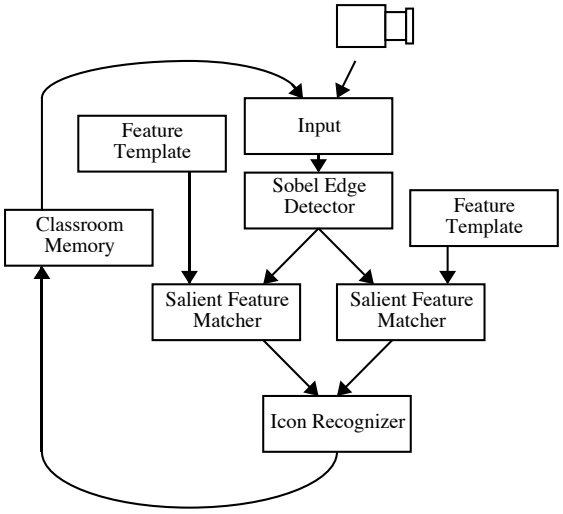

Figure 3: Icon Recognizer Pipeline

looking for known recognizable points, or *salient features*. It then recognizes specific groupings of those features as a given icon. For example, a square is an appropriate grouping of four corner features. The features are recognized using a template matcher, and then examined for any known configurations. In our current system, icons are all squares and triangles. Adding more icons is a simple matter of teaching the module the new configurations and new salient features.

Figure 3 shows the layout of a two feature icon recognizer. There is one template matching module for each type of salient feature. This allows the Classroom to recognize different icons with a simple reorganization of the existing modules. Each matcher module is passed a template to match against by the Classroom. Each template represents one of the salient features in the icon to be recognized. The templates and the region matched against can be seen in Figure 4.

Notice that the Classroom has moved the camera to get a better look at the white-board, as well as matching only against the regions where the user has written. Even something as simple as looking at the white-board shows the importance of context to a vision system. If the room did not know where the white-board was, it would have to systematically scan the entire room looking for symbols when it wanted to find the new icons. The left part of the figure is what the Classroom is looking at, the middle part is the edge image where the Classroom has told Gargoyle to attend to, and the right part of the image has the templates that are being matched.

The icon recognizer uses the relative positions of the different salient features to determine if the icon is present in the scene. Once the location and type of the hand drawn icons are known, they can be used to give the room more information. For example if you wanted to annotate the videotape that the Classroom is generating, you could use different icons to represent different annotations that the Classroom will then insert into the video.

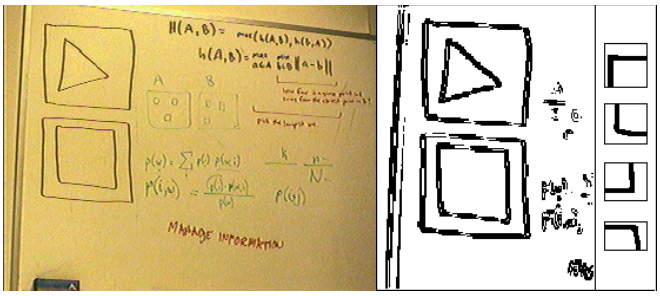

Figure 4: Template matcher input with some sample salient point templates.

The main disadvantage of this type of icon recognition is the fact that Hausdorff matching is slow. We are currently working on a number of speed improvements to our algorithm. However this is less important since the Classroom can use the context of the situation to confine the focus of attention, temporally and spatially.

FOCUSING THE ATTENTION OF INPUT MODES

One benefit of the Intelligent Classroom is that it provides a domain in which context exists and is used to leverage better vision. Focus of attention is one important way in which the Classroom leverages this knowledge. Currently the Intelligent Classroom has two ways in which to focus the attention of the visual system. One is temporally and one is spatially.

Temporally focusing the attention allows the Classroom to decide when to look at something. Some input modes are too expensive to run all of the time. For example, the Classroom can not constantly look for icons since the Hausdorff matching algorithms used by the icon recognition input mode are far too expensive to run all the time. The Classroom instead looks for new icons only when it is possible that there are new icons, such as when the user is writing on the white-board. Once the icons are recognized, they can be remembered, and do not need to be recognized again unless the user modifies them. Another way of temporally defining this action would be to have the user speak a command to the Classroom requesting it to update it's knowledge of the icons drawn on the board.

Spatial focus of attention allows the Classroom to direct the visual input mode to only look at specific locations in the scene. The Classroom provides spatial focus of attention to Gargoyle in a number of different ways. One example is instructing the input mode where to point the camera. The camera setting for tracking the person around the room is far too wide angle for reading anything off of the white board. In order to read the white-board, the camera must be focused on the white board. The Classroom is pre-calibrated with the location of the white-board, either manually or through another visual technique. This magnifies the writing enough to be seen by the camera, but this is still a large space to search for the salient points for the icons. The focus of attention can be narrowed even further by examining only that portion of the white-board on which the user has written. Figure 4 shows the Classroom eye view, looking just at the white-board after

some icons have been drawn. Figure 2 shows the Classroom watching the user write on the white-board in the normal camera configuration. The classroom's representation is on the right, and you can see that the Classroom follows the user's hand over the white-board allowing the region of interest to be cropped even more, as shown in Figure 4.

The Classroom also uses the location of the user's body parts to make the tracking faster and more accurate. By informing the "person tracking pipeline" of the location of the user, the segmenter only has to look in small portions of the image for the specific body parts. For example, the head is only going to be on top of the person, and the person won't move more than a certain amount from one frame to the next. Only by reducing the focus of attention using the context of the situation are we able to make the data that Gargoyle gives the Classroom fast and accurate.

FUTURE WORK: BEYOND FOCUS OF ATTENTION

The Classroom is still a work in progress, and there is a body of work that still needs to be done before we can test the ability of our multiple modes of input to function better than any single mode. We are optimistic, given the diverse abilities of the current input modes. By simply using some of our different input modes, without switching, we are able to interact in ways that would be impossible without those input modes. For example, it would be impossible to automatically film a lecture if the Classroom wasn't able to track the user around the room.

We are currently looking in two directions for improvement in our system. The first is allowing the Classroom to make the decisions about switching modes of input automatically for a given task. Currently the user must manually inform the Classroom of when a new input mode is needed. For our system to provide the user the full advantage of having multiple input modes, it must be able to switch them on and off seamlessly and transparently to the user.

The other direction we are actively working on is in making new input modes and improving the modes we already have. The way we hope to improve the performance of our current modes of input is by utilizing the knowledge of the Classroom. By providing the Classroom a knowledge base of how the different input modes work, the Classroom can then optimize the operation of the input modes for the given situation. We will now look at what is required for each of these improvements in turn.

Combining the Multiple Input Modes

In order to combine multiple input modes, the classroom has to make two different decisions: picking an appropriate input mode for the current situation, and deciding when to switch input modes, that is, deciding when an input mode is no longer appropriate. In order to make either of these decisions the Classroom must be able to reason about how the input modes operate. In particular it must be able to know what constraints need to hold for the mode to be appropriate. Therefore it also needs to be able to know when different constraints hold.

We are currently working on including a constraint description system into our visual input system. Whenever someone wants to add a new input mode to the system they will not only need to provide the input mode, but also a description of its constraints. These constraints will then provide the Classroom with information about when the input modes should and should not be used.

In addition to providing the constraint, the input mode developer has to provide a simple way for the Classroom to determine whether or not the constraint holds. Task constraints, that is which tasks the input mode is appropriate for, are easily checked by querying the plan recognition system about the possible tasks in which the user is engaged. Environmental context is much more difficult to determine. Sometimes it can be determined simply from the output of the input mode, but often times additional information will be needed. The constraint developer will have to include that information and tell the system where it can acquire that information. Examples of this are given in the following paragraphs.

Picking an Input Mode

There are generally two ways that the Classroom can be triggered to start a new input mode. First the user can have started a new task for which a new input mode is appropriate. Second, the constraints on another input mode may have failed while the task that required it is still active.

The first case is actually more difficult for the Classroom since many modes need to run before the Classroom can know what the environmental context is. All of the current input modes inform the Classroom whether or not their own environmental context holds. If the state of some context has not been checked recently enough (recently enough is defined depending on the context) the Classroom will simply need to run all applicable input modes, and update the current state of the environmental contexts. Additional help in picking a context can be given to the Classroom by providing it with a script detailing in which order the user will do specific tasks.

Switching to a different input mode for the same task can be easier because the same tasks often require similar constraints. This way the Classroom will have more information and can thus narrow the choices of which input mode to run without having to actually run each input mode. This implies that there needs to be a temporal aspect to the satisfaction of the constraints. Just because a constraint holds in one instance doesn't mean it will still hold at some later point.

Deciding an Input Mode is no Longer Appropriate

Since the constraints can change during the run of a given input mode for a given task, the state of the constraints need to be continuously (or occasionally) updated during the run of the input mode. This will allow the Classroom to decide whether or not the currently running input modes are appropriate. If an input mode is no longer appropriate because the task it is running for has ended, then the Classroom simply stops using that input mode. In this

case the task context no longer exists, and the Classroom does not need to run an input mode for that task.

If on the other hand the task context remains, then the Classroom will need to select an alternate input mode which suffices for the same task, but is appropriate for the new environmental contexts.

Improving Individual Input Modes

Occasionally, the environment will change in such a way that an input mode no longer functions properly, but is still the most appropriate mode. Switching which input mode you are using can be expensive, as many input modes have some initial cost. Sometimes it will make more sense to repair the mode that you are currently using rather than switch input modes.

For example, when tracking someone using the updating background subtraction system described above, constraint that the person keeps moving is broken when the person being tracked stops moving. However, the person could start moving again soon, in which case it would be incorrect to switch input modes. In this case it makes sense to modify the operation of the background subtraction system not to update the background model. This eliminates the constraint of needing the person to move, however, it adds the constraint that the background can not shift. The background tends to shift over time, so this constraint will eventually fail. The Classroom can use this time to prepare a replacement input mode.

To allow the Classroom to make modifications like this to the input modes, the constraint description system as described in the previous section is insufficient. There are two types of modifications that need to be made to the constraint description system to cover the different ways in which our visual input modes can be changed without actually switching modes. Each modification corresponds directly to a type of change that we want the Classroom to be able to perform on the input mode. The first change is simply tuning parameters to reflect small changes in the world that do not break a constraint, but do degrade performance. The second is a major change: the need to compensate for some change in the environment that breaks one of the input mode's constraints. We will now look at what modification of the constraint description system these abilities will require.

Tuning parameters is actually very similar to focusing attention. Parameters are just continuous values which affect the operation of the input modes. The classroom needs to know two things: which direction has what effect, and how large the changes need to be to have certain effects. As an example, consider the threshold of an edge detector whose output is used for a template matcher. If the threshold is too high, no edges will make it through, if it is too low, there will be too much noise to find the salient edges. This gives the Classroom two constraints that might fail which it can monitor. If it can no longer detect salient edges, there might be a problem with the edge threshold. In order to solve this the Classroom can sample the edge density to see which direction it needs to be changed. If the density is too high, the threshold must be raised, and vice versa. The designer of the input mode will unfortunately need to do experimentation to determine a translation from edge density to a change in threshold. This is not that a large cost compared to the amount of work that normally goes into tuning systems to work in specific contexts. Once this relationship has been determined the Classroom can simply monitor the operation of different modules in a given input mode, and update the parameters as needed.

Actually changing the method of operation of a given input mode can often be easier for the Classroom than simply adjusting parameters, since it only needs to know when to switch and does not need to calculate values for the change. To know if the change needs to happen is very simple, just watch for a constraint failure, just like switching from one mode to another. This time though, the constraint should allow the Classroom to modify the input mode's operation rather than just terminating that input mode. The alternate operation mode will then introduce new constraints which must also hold if the operation mode switch is valid. If no operational mode is valid for that input mode the input mode is no longer appropriate and the Classroom will need to select a new one. In order to better understand how these techniques work we will now look at a short example.

Background Subtraction Example

In this example we will look at some violations of the constraints on the Background Subtraction input mode.

The Classroom has been tracking the user for some time using the background subtraction input mode. While doing this, the Classroom surreptitiously acquired a color histogram of the person which it stores away for later use. In the event of a drastic violation, like a second person entering the scene, the entire input mode is made invalid and needs to be replaced. The Classroom is monitoring the output of the background subtraction input mode, and notices when the morphological segmenter returns two objects rather than one person. Since the tracking task is most likely still needed, it needs to be replaced with another person tracking input mode. The only alternate tracking method currently available is the color histogram input mode. By monitoring the output of the person tracker, the Classroom able to prevent a major breakdown when one of the main constraints is broken.

The Classroom is also able to handle smaller violations by modifying the input modes through parameters of the individual modules. The Classroom has control over all of the parameters for all of the modules in Gargoyle, such as how much a point can change before it is no longer considered to be background. Another interesting control that the Classroom has over the background module is whether or not the background is updated. Under normal operation, the background is updated each time through the control loop. This allows the background to "learn" systematic lighting changes to the scene, for example the lights being turned on; and additions to the background, for example the user moving something in the scene. If the lighting changes, the Classroom is informed of a massive

shift in the number of candidate pixels from one frame to the next. It can then adjust the width of the distance to the mode for background pixels. This reduces the quality of the tracking, but allows the background to reacquire, while continuing to acquire reasonably accurate tracking information. Another way the room can improve the performance of the background subtraction is to use its knowledge of the movement of the user. If the user stands in one place for too long, he will fade into the background and be lost by the tracker. In order to prevent this from happening, the Classroom will realize when the user has stopped moving, and will stop the background from updating, resuming the updating again when it is possible and necessary.

These examples show a number of constraints that the background subtraction module relies on to provide good results. The only way for it to provide robust results is to use the Classroom's knowledge of the situation. The only way that it can remain effective when the person stops moving, is to use the knowledge that the person has stopped moving, and stop updating the background.

RELATED WORK

The vision work in the Intelligent Classroom has been influenced by a number of different sources. Much of the infrastructure for Gargoyle, and the notion of the needs of a real time computer vision system were developed for the active vision system on the University of Chicago's robot [3,8]. We were inspired to create small fast vision routines that rely on specific context by earlier successful work in robotics [7]. Finally our reliance on goal based context comes from a wealth of active vision research especially [11].

Other research groups are working on similar tasks. Pfinder tracks people in a similar manner, and uses additional cues to give better locality to the person [13]. We hope to build up similar broader models of the user in the future, but rather than use all of the cues, we plan to only use the most accurate cues at any given moment. The Zombie board is a system for recognizing symbols on a white-board [10], but the user must write on the board to interact with the icons. The filming task is similar to Bobick's work in automatically filming TV shows [9]. While our video quality is lower, we require no user intervention.

It is also worthwhile to note that Wasson et al have been working on similar execution and sensing problems as shown in [12].

CONCLUSION

This paper has shown some of the work that we have done on interacting with the Intelligent Classroom using multiple methods of sensing the environment. By carefully utilizing computational resources we hope to enable a new level of interactivity in intelligent environments.

REFERENCES

1. Aloimonos, Y. Purposive and qualitative active vision. In *Proceedings on the International Conference on Pattern Recognition*, pages 346-360, 1990
2. Chapman D., *Vision Instruction and Action*. MIT Press, 1991.
3. Firby, R. J., Kahn, R. E., Prokopowicz, P. N., and Swain, M. J. An architecture for vision and action. In *Proceedings on the International Joint Conference on Artificial Intelligence,* 1995
4. Flachsbart, J. Gargoyle: Vision in the Intelligent Classroom. Master's thesis, University of Chicago, 1997.http://www.cs.uchicago.edu/~josh/official_content/Papers.html
5. Franklin, D., and Flachsbart, J. All gadget and no representation makes Jack a dull environment. Technical Report SS-98-02, American Association for Artificial Intelligence, 1998.
6. Franklin D. Cooperating with people: the Intelligent Classroom. In *Proceedings of the AAAI98* (Madison WI, July 1998), MIT Press, pages 555-560.
7. Horswill, I. *Specialization of Perceptual Processes.* PhD Thesis, Massachusetts Institute of Technology, 1993.
8. Kahn, R. E. *Perseus: An Extensible Vision System for Human Machine Interaction.* PhD Thesis, University of Chicago, August 1996.
9. Pinhanez, C., and Bobick, A. Intelligent Studios: Using computer vision to control TV cameras. *IJCAI Workshop on Entertainment and AI/Alife*, April 1995.
10. Saund, E. 1996. Machine interpretation of diagrammatic user interfaces to office whiteboard appliances. Technical Report, Xerox PARC, 1996.
11. Strat, T. M., *Natural Object Recognition.* Springer-Verlag, 1992.
12. Wasson G., Kortenkamp, D., Huber E. Integrating Active Perception with an autonomous robot architecture. In *Proceedings on the Second International Conference on Autonomous Agents* (Minneapolis MN, May 1998), ACM, pages 325-331
13. Wren, C., Azarbayejani, A., Darrell, T., and Pentland A. Pfinder: Real time tracking of the human body. Media Lab Tech Report 353, Massachusetts Institute of Technology, 1995.