VICs: A Modular HCI Framework Using Spatio-Temporal Dynamics

Guangqi Ye, Jason J. Corso, Darius Burschka, and Gregory D. Hager

Computational Interaction and Robotics Laboratory
The Johns Hopkins University
{grant|jcorso|burschka|hager}@cs.jhu.edu

Abstract. Many vision-based human-computer interaction systems are based on the tracking of user actions. Examples include gaze-tracking, head-tracking, finger-tracking, and so forth. In this paper, we present a framework that employs no user-tracking; instead, all interface components continuously observe and react to changes within a local neighborhood. More specifically, components expect a pre-defined sequence of visual events called visual interface cues (VICs). VICs include color, texture, motion and geometric elements, arranged to maximize the veridicality of the resulting interface element. A component is executed when this stream of cues has been satisfied.
We present a general architecture for an interface system operating under the VICs-based HCI paradigm, and then focus specifically on an appearance-based system in which a Hidden Markov Model (HMM) is employed to learn the gesture dynamics. Our implementation of the system successfully recognizes a button-push with a 96% success rate.

1 Introduction

The promise of computer vision for human-computer interaction (HCI) is great: vision-based interfaces would allow unencumbered, large-scale spatial motion. They could make use of hand gestures, movements or other similar input means; and video itself is passive, (now) cheap, and (soon) nearly universally available. In the simplest case, tracked hand motion and gesture recognition could replace the mouse in traditional applications. But, computer vision offers the additional possibility of defining new forms of interaction that make use of whole body motion, for example, interaction with a virtual character [19].

A brief survey of the literature (see Section 1.1) reveals that most reported work on vision-based HCI relies heavily on visual tracking and visual template recognition algorithms as its core technology. While tracking and recognition are, in some sense, the most popular direction for developing advanced vision-based interfaces, one might ask if they are either necessary or sufficient. For example, complete, constant tracking of human body motion might be a convenient abstraction for detecting that a user's hand has touched a virtual "button," but general human motion tracking is well-known to be a complex and difficult problem [23,9]. What if button contact can instead be detected using simple

motion or color segmentation combined with template matching? On the other hand, while template matching is a relatively well-understood mechanism for recognizing static configurations, can it be similarly effective when accounting for the inherent spatio-temporal dynamics of human motion? What if, as in most cases, the user is not interacting at all? Is there any reason to perform potentially expensive image processing to generate negative results? Finally, one might turn the question around and ask whether it is possible to add structure to the interaction problem so as to render the vision problem solvable with high reliability.

The goal of this paper is to present visual interface cues (VICs), a general and extensible paradigm that addresses many of these issues. In Section 2, we outline a general approach to the development of modular, spatially cued vision-based interaction, and describe prior systems that make use of this basic idea. In Section 3, we turn to the question of modeling local spatio-temporal gesture dynamics using Hidden Markov Models. In Section 4, we present results using HMM-based event recognition for a simple push-button VICon. Finally, we describe our current work on extending and applying the VICs paradigm.

1.1 Related Work

The Pfinder system [32] and related applications [19] is a commonly cited example of a vision-based interface. Pfinder uses a statistically-based segmentation technique to detect and track a human user as a set of connected "blobs." A variety of filtering and estimation algorithms use the information from these blobs to produce a running state estimate of body configuration and motion [31]. Most applications make use of body motion estimates to animate a character or allow a user to interact with virtual objects.

More broadly, from the point of view of vision, there has been a great deal of interest in tracking of human body motion, faces, facial expression, and gesture, e.g. [2,12,25,5,33,10,3,21,18,8,4], with the general goal of supporting human-computer interaction.

From the HCI perspective, there have also been a wide class of "demonstration systems" that make use of vision as their input. The ZombiBoard [20] and BrightBoard [26] are examples of extensions of classical 2-D "point-and-click" style user interfaces to desktop/blackboard style interactions. They allow, for example, the selection, capture, or manipulation of items viewed by a video camera on a whiteboard or desktop. Input is usually via special written tokens; vision processing is based on simple background subtraction or thresholding followed by binary image processing, much as with Pfinder. More extensive proposals for mixing virtual and physical documents on the desktop include work on the Digital Desk [30] and on the "office of the future" [22]. A good example of a gesture-based interface is GestureVR [25].

2 The VICs Paradigm

2.1 Modeling Interaction

Current interface technology, Windows-Icons-Menus-Pointers (WIMP) [29], is modeled with a simple state-machine (Figure 1). The dominant interface component in these *third-generation interfaces* is the icon. Typically, these icons have one pre-defined action associated with them that is triggered upon a mouse click.

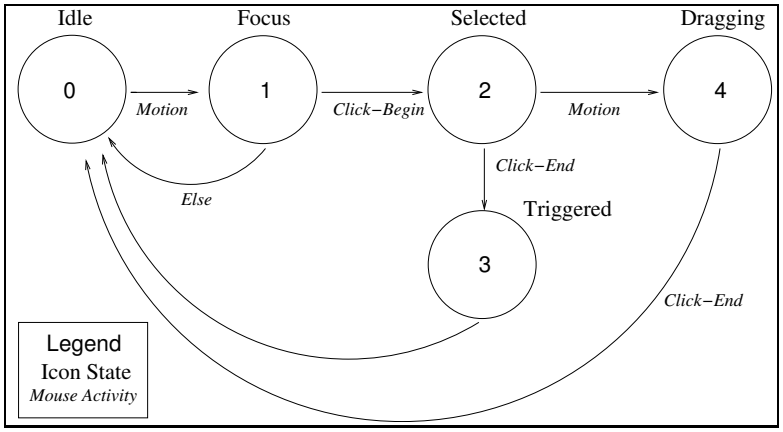

Fig. 1. The icon state model for a WIMP interface.

We extend the functionality of a traditional icon by increasing the number of associated actions that can be triggered by the user in a straightforward manner that does not require the user to learn complicated interaction semantics. For standard WIMP interfaces the size of this set is 1: point-and-click. We call a *super*-WIMP interface one that includes multi-button input or mouse-gesture input. One such example is the SKETCH framework [34]. For the *super*-WIMP interfaces the size of this set is larger, but still relatively small; it is limited by the coarse nature of mouse input. Our vision-based extension greatly increases the set of possible user inputs by employing the increased spatial input dimensionality. To allow for such an extension, the notion of the icon must change: we define a VICs-based interface component (VICon) to be composed of three parts. First, it contains a visual processing engine. This engine is the core of the VICon as it replaces the current point-and-click nature of third-generation interfaces. Second, it has the ability to display itself to the user, and lastly, it has some application specific functionality.

As mentioned earlier in Section 1, the VICon does not rely on tracking algorithms to monitor the user and detect actions. Instead, the VICon watches a region-of-interest (ROI) in the video stream and waits for recognizable user-input. For instance, if we model a simple push-button, the VICon might watch for something that resembles a human-finger in its ROI.

The obvious approach to detect user interaction is one of template-matching in which the VICon is aware of a set of possible gestures and uses image processing techniques to analyze the ROI in every frame of video. However, in practice, such a method is prone to false-positives by spurious template matches. Also, a template matching approach, alone, is potentially wasteful because it is more expensive than other simpler tasks like motion detection and color segmentation that may easily indicate a negative match.

If one observes the sequence of cues that precede a button-push, for instance, one notices that there are distinct cues comprising the actual button push: motion, color-blob, edges. This sequence of cues, ordered from simple to complex, can be used to facilitate efficient, accurate user-input detection. Define a *selector* to be a vision component that computes some measure on a local region of an image, and returns either nothing, indicating the absence of a cue or feature, or values describing a detected feature [13]. For example, a motion selector might return nothing if there is no apparent image motion or a description of the size and magnitude of a region of detected motion. Thus, at its core, the visual processing engine of the VICon is a sequence of selectors: we call it a *visual interaction cue parser* or just a parser.

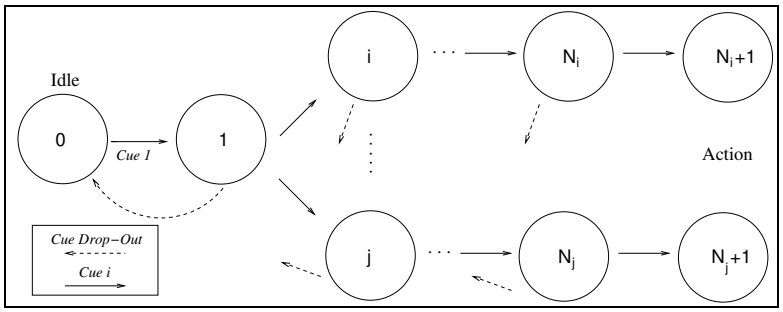

Fig. 2. The state model for a VICs-based interface component.

Formally, we define a visual interaction cue parser (Figure 2). It is a component with the following structure:

1 A finite set of discrete states $s_1, s_2, ...s_n$.
2 A distinguished initial state s_1.
3 Associated with each state s_i, a function f_i on the incoming input stream that defines a continuous state variable x.
4 For each state s_i, a set of transition rules that associates an event $e_{i,j}$, $j = 1 \ldots m \leq n$ (informally, the output of a selector) with either a state of higher index, or s_1. By convention, the first transition event to fire defines the transition for that state.

We return to the example of a button push from above. Using a parser, we create a possible sequence of selectors: (1) a simple motion selector, (2) a coarse

color and motion selector, (3) a selector for color and cessation of motion, and (4) gesture recognition. It is easy to see that processing under this framework is efficient because of the selector ordering from simple to complex wherein parsing halts as soon as one selector in the sequence is not satisfied. More powerful and sophisticated parsing models are plausible under this paradigm: an example showing the use of a Hidden Markov Model is presented in Section 3.

The intent of the framework is that a parser will not only accept certain input, but might return other relevant information: location, duration. The key factor differentiating the VICs paradigm from traditional interface components is that there may be multiple exit cases for a given VICon determined by different streams through the parser each triggering a different event. The lexicon of possible triggers is an order of magnitude larger than WIMP and *super*-WIMP interfaces.

2.2 Interaction Modes

The notion of a VICs-based interface is broad and extensible to varying application domains. In this section we enumerate the set of interaction modes in which a VICon may be used.

1. **2D-2D Projection** - Here, one camera is pointed at a workspace, e.g. tabletop. One or many projectors are used to project interface components onto this surface while the video-stream is processed under the VICs paradigm. This mode has been proposed in [35]. We feel incorporating VICs-based interface components will increase its effectiveness and broaden the domain of applications.
2. **2D-2D Mirror** - In this mode of interaction, one camera is aimed directly at the user and the image stream is displayed in the background of the user-interface for the user. Interface components are then composited into the video stream and presented to the user. This interface mode could also be used in a projection style display to allow for a group to collaborate in the shared space. Figure 4 shows some example applications of this model.
3. **3D-2D Projection** - This mode is similar to the first (2D-2D Projection) except that 2 or more cameras will be aimed at the workspace and the set of possible selectors is increased to include more robust 3D geometry. In Figure 3 the prototype 4D-Touchpad system is shown. The 4D-Touchpad is an experimental platform based on the VICs framework [7].
4. **2.5D Augmented Reality** - Both video-see-through and optical-see-through augmented reality are possible if the user(s) wear stereo head-mounted displays (HMD) [1]. With stereo cameras mounted atop the HMD, knowledge of a governing surface can be extracted from the view, e.g. planar surface [6]. All VICons can then be defined to rest on this governing surface and interaction is defined with respect to this surface. One possible application is a piano where each key is a separate VICon.
5. **3D Augmented Reality** - In this case, we remove the constraint that the interface is tied to one governing surface and allow the VICons to be

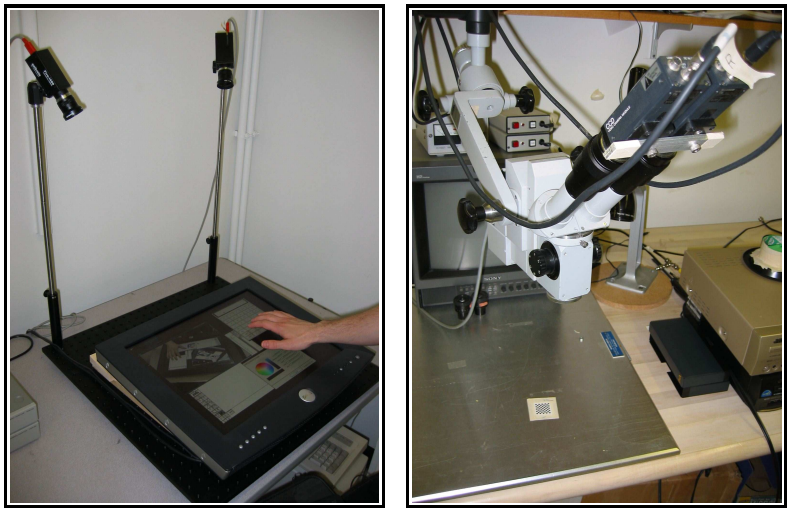

Fig. 3. (left) The prototype of the 4D-Touchpad using a flatpanel as the projector for the display. (right) A 3D surgical interaction system under development based on the VICs framework.

fully 3D. Two example applications areas are (1) motor-function training for young children and (2) medical applications. Shown in Figure 3 is a stereo microscope that is being used as a VICs-based surgical environment.

2.3 Prior State of the Art in VICs Technology

In this section we show a small set of example interfaces built under the VICs paradigm: interaction through a stream of local-based selectors. First, we show a simple button-based VICon in a calculator setting (Figure 4-left). In this case, the VICon used a motion-based cue, a color-segmentation cue, and enforced that the color remain present for a static time-interval. Next, we show multiple triggers based on user-input (Figure 4-middle). Here, the user can select the ball, drag it, and release. The parser incorporates a simple-gesture recognition stage; it's state-model follows Figure 2. As mentioned earlier, motion and dynamics can be added to the VICons. Figure 4-right shows a *Breakout*^{TM} like program where the ball is a VICon. During play, the ball (the VICon) travels through the workspace. The user attempts to prevent the ball from falling through the bottom of the workspace while deflecting it toward the colored bricks at the top of the workspace; notice the VICon is not anchored.

The thprevious three VICs-based interface examples employ the 2D-2D mirror mode of operation. Our current focus is the 2.5D augmented reality mode of operation. To this end, we have developed a set of fast surface recovery techniques [6] allowing us to anchor the interface to a planar surface.

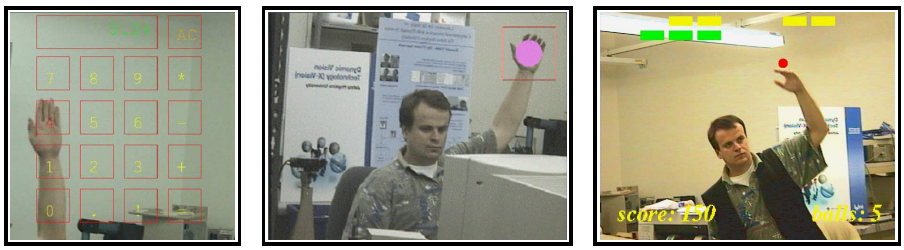

Fig. 4. (left) A VICs-based calculator using a motion-color parser. (middle) Gesture-based demonstration of multiple interaction triggers for a single VICon. (right) VICs-based 2D-2D mirror mode interface for a *Breakout*^{TM} style game.

2.4 Robustness and Applicability

In this section, we presented a general framework for vision-based interaction that provides set of efficient techniques to employ video as input to computer systems without requiring the user to learn complex interaction semantics. We argue that given a finite set of interaction primitives, the site-centric parsing will perform well. Under the site-centric model, each interface object is *trained* on a prior set of activation patterns through its parser. These selectors are built with well understood image processing techniques and operate on a small subset of the image. In Section 4, we provide experimental results that support our claim of the scheme's robustness.

While in its current state the topology of the parser and implementation of the selectors is the task of the developer, we are investigating learning methods (e.g. neural networks) to make this process automatic. The set of example interfaces we present in this paper demonstrate that this is not a hindrance to the adoption of the VICs framework.

The VICs model can immediately be employed to provide fully-functional standard user interfaces by mimicking their components (buttons, scrollbars, etc). However, it is not universally applicable for there may be some interfaces perfectly suited to user-centric interaction models. For example, eye gaze tracking may be the only way to enable interaction during automobile driving.

3 A Stochastic Extension of the Parser

As noted at the outset, one of the central problems in vision-based interaction is to effectively manage the complexity of the input space. As a first step, VICs makes use of geometrically localized interaction and state-based activation to minimize both input complexity and computational complexity. As a next step, we are abstracting the visual input space using ideas originally developed in the area of speech processing (Figure 5).

In a speech system, interpretation of the incoming waveform is performed in two steps. First, an acoustic pre-processor turns the continuous input wave-

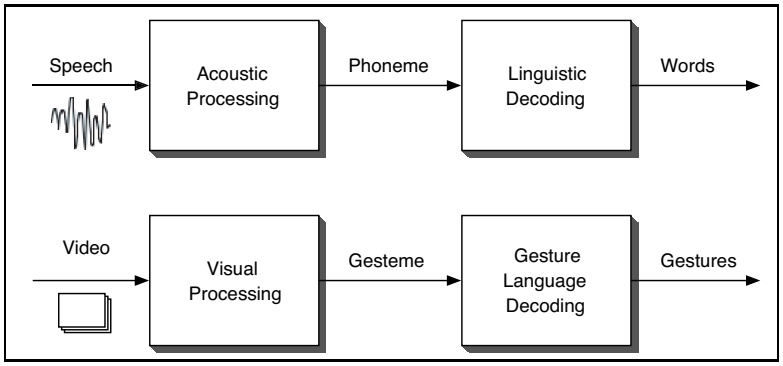

Fig. 5. The parallels between a speech processing system and visual "language" processing.

form into a relatively small set of discrete "symbols" (phonemes). Subsequently, temporal sequences of phonemes are processed into words using a learned language model. In our proposed vision processor, we also abstract the incoming high-dimensional image sequence into a lower-dimensional discrete input string, now representing some aspect of spatial appearance, and likewise train temporal models on these "gestemes" to recognize complete gestural cues.

In the remainder of this section, we describe the results of an initial prototype VICs-based interaction system to identify a button-pushing action. We use a static camera to supervise a virtual button, which is represented by a graphical icon. The user is expected to move his finger toward the button and stay on the button for a short period of time to trigger it. The system will decide if the user has triggered the button. Thus, fast and robust foreground segmentation and action recognition are two key elements of our system.

3.1 Background Modeling and Image Segmentation Based on Hue Histograms

Background subtraction, gray-scale background modeling [14], color appearance modeling [28], color histogram [17] and combining of multiple cues [24] are among the most widely used methods to model the background and perform foreground segmentation. We propose to use a hue histogram for its computational efficiency and relative color invariance. Hue is a good color-invariant model that is relatively invariable to translation and rotation about the viewing axis, and changes slowly under change of angle of view, scale and occlusion [11].

We assume that the background is known for a given session. We split the background image into an array of non-overlapping equal-sized sub-images. For each sub-image, we build a hue histogram to model it. We process the foreground image in a similar way and perform pairwise histogram matching between background and foreground image histograms. Here, we employ histogram intersection [28] as the comparison criterion.

$$H(F,B) = \frac{\sum_{j=1}^{n} min(F_j, B_j)}{\sum_{j=1}^{n} B_j} \quad (1)$$

Here B and F refer to the background and foreground histogram respectively. If the matching value is below the threshold, which is determined empirically, the corresponding sub-image is classified as foreground; otherwise, it is background. Our experiments show that combining a hue color model with histogram intersection can achieve relative invariance to illumination changes and obtain good segmentation results. Figure 6 shows an example.

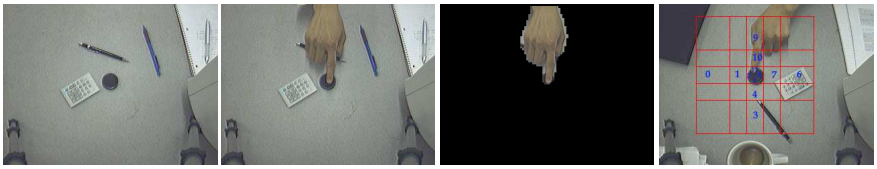

Fig. 6. An example of background image, unsegmented image and segmentation result. The leftmost image is the background. The second image shows when the hand has entered the scene. The final segmentation result is shown in the third image. The last image demonstrates our feature space definition.

3.2 HMM-based Human Activity Recognition

In our experiment, we employ HMM [15] techniques [16] to train and recognize the button-pushing action. The basic idea is to define a finite feature space onto which the image is mapped. Given any captured image sequence, we convert it to an array of discrete features. A discrete forward HMM is constructed for each class of actions and trained based on training sequences using the Baum-Welch algorithm [16]. The probability that each HMM generates the given feature sequence is the criterion of recognition.

We propose a computationally efficient and robust feature extraction scheme. The extracted feature indicates the direction and distance of the finger from the center of the button. In principle, we split the local activation region of the button into a 5 by 5 grid. Based on the foreground segmentation result, we can claim whether a certain cell is covered by the hand by thresholding the percentage of the foreground pixels in the cell. The direction of the hand with respect to the button is then decided by comparing the number of cells touched by the hand in each of the four directions. The distance of the finger with respect to the button is calculated as the distance between the center of the button and the nearest cell covered by the hand. Combination of all discretized directions and distances forms our feature space.

We experiment with two different HMM structures to model the dynamics of the gestures. There are four different gestures, each of them modeling triggering the button from a distinct approaching direction: i.e. left, right, up and down. The first HMM paradigm is the singleton-based composite model. For each feature state, we define a basic singleton HMM (Figure 7) to represent the dynamics of the finger movement for this particular spatial configuration. To capture the temporal dynamics of the whole gesture, we learn a representative sequence for each of the four classes of actions. Based on this characteristic sequence, we build the HMM for this class by concatenating, with null transitions, all the basic singleton HMMs corresponding to each symbol in the sequence. Figure 7 shows the structure of this singleton-based HMM structure. In essence, this topology models the gesture as a tour through a series of spatial configurations.

The second model is a three state forward HMM as shown in Figure 7. Each class shares the same topology, with the parameter set trained separately based on the sequences belonging to corresponding class. The attractiveness of this topology is that it intuitively models the temporal dynamics of the gesture, as opposed to the singleton-based HMMs that try to model each of the symbols in the observation dictionary. In our experiment, for each of the four classes, we expect the user to move the finger through the feature space in the following order: outer layer, inner layer and center of the button. Intuitively, the dynamics is composed of three distinct stages. Ideally, each of the three states of the forward HMM would model the corresponding stage of the dynamics. The parameter sets of the trained class HMMs verify our expectation. For example, if the recognizer observes a sequence in which the hand stays on the button all the time but does not witness the process showing the hand approaching the button, it will reject the sequence as a valid trigger action. In general, a dynamic process with n stages can be modeled using an n-state forward HMM with similar topology. For example, in [27], four-state HMMs are used to recognize American Sign Language.

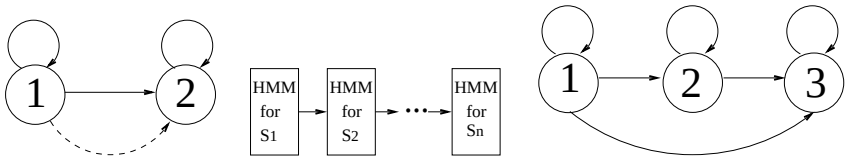

Fig. 7. HMM structures used to model gestures. The left figure shows the structure of a basic singleton HMM. The center figure illustrates the composite HMM topology that concatenates singleton HMMs. The topology of the simple three-state HMM is shown in the right figure.

Since it is difficult to capture all possible patterns of non-pushing actions, we use a threshold on the highest possibility of the classes to perform rejection. However, the duration of the action and the possibility that each class generates

such a sequence may vary significantly even though the action pattern is still the same. To overcome this time variation, we perform sequence aligning in training and recognition. That is, we choose a fixed length to be the standard length (for example, 20 frames). We perform sampling or interpolation for longer or shorter sequences to fit our standard.

4 Experimental Results

In our current experiment, we use a color camera with an image size of 640×480 as the imaging sensor on a standard Pentium III 1Ghz PC running Linux OS.

4.1 Background Modeling and Segmentation Result

To test our segmentation scheme, we captured image pairs of the background and foreground. By comparing the segmentation result and the ground-truth classification image, which is generated by manually marking the foreground part of the scene, we are able to evaluate this algorithm. We captured more than 20 pairs of background/foreground images with different background scenes and carried out the experiment on these images. The test set also includes 6 pairs of images that undergo illumination changes. As a result, the average correct ratio based on per pixel basis is 98.16%, with average false positive ratio of 1.55% and false negative ratio of 0.29%. Figure 6 shows a typical segmentation example.

As shown in Figure **??**, we also compare the segmentation result with different sub-window sizes and with different numbers of hue histogram bins. The result shows that histograms with at least 8 bins perform better than those with less, while increasing the bins to 16 or more does not bring much performance enhancement for our experimental set. It can be foreseen that more bins will be needed to carry out good segmentation for those images in which the appearance of the foreground is close to that of the background. The result also shows that the false positive ratio will decrease with the increment of tile size because imaging noise would be compressed when the histogram is constructed over a larger neighboring area, which in essence is a process of averaging the appearance of the tile.

4.2 Action Recognition Result

For training and testing of our HMMs, we recorded over 300 action sequences performed by 6 different people, with 76 of the sequences being used for training. For the singleton-based model, an offline procedure is carried out to find the best characteristic sequence for each class and construct the HMM. For both the singleton-based model and the simple 3-state forward HMM, we carry out training. Both models achieve correct ratio of 100% on the training set. We also compare them by recognizing a test set of over 270 sequences. The length of the test sequences varies significantly, ranging from 30 to over 200. The test set includes some sequences with illumination changes, which are also segmented

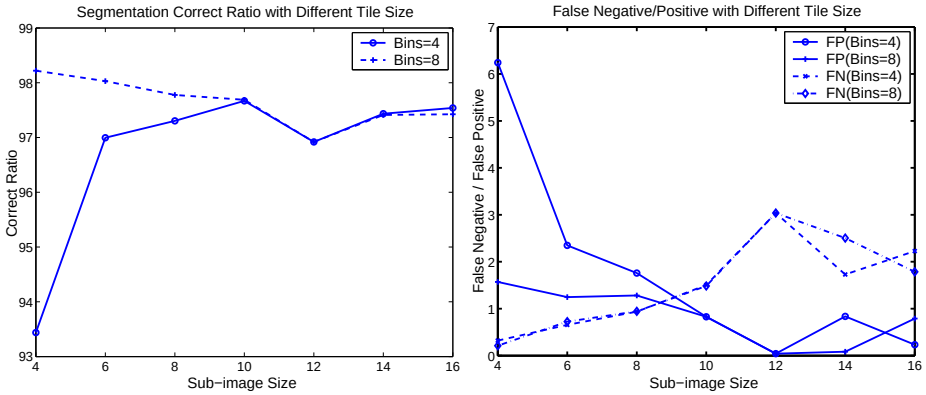

Fig. 8. Segmentation correct ratio/false positive/false negative with different sub-image size and number of bins in the histogram.

successfully. To test the rejection scheme, we also include invalid sequences in the test set, such as those that do not conform to the temporal pattern of the model gesture and those that do not follow the correct order of expected visual features of the models. The resulting correct ratio is 96.8% for the singleton-based HMM model and 96.0% for 3-state simple HMM. This suggests that a simple 3-state HMM can model the button-pushing gesture almost as good as a 60-state (standard length(=30) $\times$ 2) singleton-based composite HMM.

For the singleton-based model, the standard length of the class characteristic sequence will influence the system performance and speed. Along with the increase of the size of primary sequence, the time needed to carry out the recognition will also grow linearly. However, since a longer sequence contains more information and thus, has a larger HMM, the total system performance will improve. The following table shows the experimental results with differing standard lengths.

Table 1. Experiment results with different length of characteristic sequence

L	Accuracy of Training Set	Accuracy of Test Set
10	100.0%	86.8%
20	100.0%	94.2%
30	100.0%	96.8%

For the simple n-state forward HMM structure, we also carry out experiments with different number of states in the topology of the HMM. It is not surprising to find an increase in the number of states from 3 to 4 or 5 doesn't bring much accuracy improvement for our simple 3-stage gesture. The result supports our

previous claim that an n-state HMM can model a dynamic process with n stages well.

5 Conclusion

We have introduced the VICs approach to vision-based interaction. VICs stems from our experience using locally activated iconic cues to develop simple vision-driven interfaces. In particular, we have identified two central problems to be solved: (1) developing reliable foreground-background disambiguation and (2) incorporating dynamics into gestures. We have shown that, given good solutions to the former problem, the latter can be addressed using standard HMM techniques.

Our immediate goal for the VICs project is to create 2.5D surface-anchored interfaces. To this end, we have developed a set of fast surface recovery techniques to place two rectified images in correspondence [6], and we are currently extending the results reported in this paper to a two-camera system. In the latter case, the HMM input will be data from both images, and the goal will be to recognize that the user is pressing a button as if it appears on the underlying surface.

Currently, we use general HMMs to train and recognize different gestures. However, other techniques, such as an HMM based on minimum classification error and duration-HMMs, could help to improve the recognition performance of the system. We are also interested in expanding our gesture set to allow richer and more complex interaction, such as triggering a switch, twisting a dial, dragging an icon, etc.

References

1. R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6:355–385, 1997.
2. S. Basu, I. Essa, and A. Pentland. Motion regularization for model-based head tracking. In *Proc. Int. Conf. Pattern Recognition*, 1996.
3. M.J. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. *Int. J. Computer Vision*, 25(1):23–48, 1997.
4. G. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, April 1998.
5. C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. Computer Vision and Pattern Recognition*, pages 8–15, 1998.
6. Jason J. Corso, Darius Burschka, and Gregory D. Hager. Direct Plane Tracking in Stereo Image for Mobile Navigation. In *Proceedings of International Conference on Robotics and Automation*, pages 875–880, 2003.
7. Jason J. Corso, Darius Burschka, and Gregory D. Hager. The 4DT: Unencumbered HCI With VICs. In *Proceedings of IEEE Workshop on Computer Vision and Pattern Recognition for Human Computer Interaction*, 2003.
8. Y. Cui and J. Weng. View-based hand segmentation and hand-sequence recognition with complex backgrounds. In *ICPR96*, page C8A.4, 1996.

9. D. Gavrila. The visual analysis of human movement: a survey. *Computer Vision and Image Understanding*, 73:82–98, 1999.
10. D. Gavrila and L. Davis. Towards 3-d model-based tracking and recognition of human movement: A multi-view approach. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, 1995.
11. Theo Gevers. Color based object recognition. *Pattern Recognition*, 32(3):453–464, 1999.
12. L. Goncalves, E. Di Bernardo, E. Ursella, and P. Perona. Monocular tracking of the human arm in 3-d. In *Proc. Int. Conf. Computer Vision*, pages 764–770, 1995.
13. G. Hager and K. Toyama. Incremental focus of attention for robust visual tracking. *International Journal of Computer Vision*, 35(1):45–63, November 1999.
14. Thanarat Horprasert, David Harwood, and Larry S. Davis. A robust background substraction and shadow detection. In *Proc. ACCV'2000, Taipei, Taiwan*, January 2000.
15. K. Ishii J. Yamota, J. Ohya. Recognizing human actions in time-sequential images using hidden markov model. In *IEEE Proc. CVPR 1992, Champaign, IL*, pages 379–385, 1992.
16. Frederick Jelinek. In *Statistical Methods for Speech Recognition*, MIT Press, 1999.
17. Michael J. Jones and James M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46(1):81–96, 2002.
18. R. Kjeldsen and J.R. Kender. Interaction with on-screen objects using visual gesture recognition. In *CVPR97*, pages 788–793, 1997.
19. P. Maes, T.J. Darrell, B. Blumberg, and A.P. Pentland. The alive system: Wireless, full-body interaction with autonomous agents. *MultSys*, 5(2):105–112, March 1997.
20. T. Moran, E. Saund, W. van Melle, A. Gujar, K. Fishkin, and B. Harrison. Design and technology for collaborage: Collaborative collages of information on physical walls. In *Proc. ACM Symposium on User Interface Software and Technology*, 1999.
21. V.I. Pavlovic, R. Sharma, and T.S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *PAMI*, 19(7):677–695, July 1997.
22. R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *Proc. SIGGRAPH*, 1998.
23. J.M. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: An application to human hand tracking. In *Computer Vision – ECCV '94*, volume B, pages 35–46, 1994.
24. Christopher Richard Rwen, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7)*, 19(7):780–784, 1997.
25. J. Segen and S. Kumar. Fast and accurate 3d gesture recognition interface. In *ICPR98*, page SA11, 1998.
26. Quentin Stafford-Fraser and Peter Robinson. Brightboard: A video-augmented environment papers: Virtual and computer-augmented environments. In *Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems*, pages 134–141, 1996.
27. T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. Technical Report TR-375, M.I.T. Media Laboratory, 1996.
28. M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision 7(1)*, pages 11–32, 1991.
29. Andries van Dam. Post-wimp user interfaces. *Communications Of The ACM*, 40(2):63–67, 1997.

30. Pierre Welner. Interacting with paper on the digital desk. *Communications of the ACM*, 36(7):87–96, 1993.
31. C. Wren and A. Pentland. Dynamic modeling of human motion. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, 1998.
32. C.R. Wren, A. Azarbayejani, T.J. Darrell, and A.P. Pentland. Pfinder: Real-time tracking of the human body. *PAMI*, 19(7):780–785, July 1997.
33. M. Yamamoto, A. Sato, and S. Kawada. Incremental tracking of human actions from multiple views. In *Proc. Computer Vision and Pattern Recognition*, pages 2–7, 1998.
34. Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. Sketch: an interface for sketching 3d scenes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 163–170. ACM Press, 1996.
35. Zhengyou Zhang, Ying Wu, Ying Shan, and Steven Shafer. Visual panel: Virtual mouse keyboard and 3d controller with an ordinary piece of paper. In *Workshop on Perceptive User Interfaces*. ACM Digital Library, November 2001. ISBN 1-58113-448-7.