Vision-based Interfaces for Mobility

Mathias Kölsch Matthew Turk Tobias Höllerer James Chainey
Department of Computer Science
University of California
Santa Barbara, CA 93106

Abstract

This paper introduces fast and robust computer vision methods for hand gesture-based mobile user interfaces. In combination with other algorithms, these methods achieve usability and interactivity even when both the camera and the object of interest are in motion, such as with mobile and wearable computing settings. By means of a head-worn camera a set of applications can be controlled entirely with gestures of non-instrumented hands.

We describe a set of general gesture-based interaction techniques and explore their characteristics in terms of task suitability and the computer vision algorithms required for their recognition. By doing so, we present an arsenal of mostly generic interaction methods that can be used to facilitate input to mobile applications. We developed a prototype application testbed to evaluate our gesture-based interfaces. We chose three basic tasks from an infrastructure maintenance and repair scenario to illustrate the applicability of our interface techniques.

1. Introduction

Vision-based interfaces (VBI) in stationary installations have recently achieved a quality level acceptable to consumers for a number of compelling applications. For example, Sony's Eye Toy, an accessory for the PlayStation 2, has enjoyed unexpected commercial success and topped the UK game sales charts for months. A USB camera recognizes the players' full-body motions and projects the player directly into the game. Mobile computer vision systems, however, have not seen the same level of maturity. Their lack of robustness, speed, and accuracy previously prevented reliable interface operation when both the camera and the object of interest are moving. We demonstrate in this paper that hand gesture recognition by means of computer vision methods is both feasible and advantageous in mobile computing environments.

User input to a mobile or wearable computer is possible using a number of modalities: Through a device with a receptive surface (PDA touch-screen, hand-held chording keyboard, phone keypad), through sensors worn or attached to the body (data gloves, orientation sensors), through speech recognition (speaker phone with voice activation), or through non-contact computer vision (CV) methods. For every technology there are environmental conditions in which it is favorable to use alternative technologies. For example, physical input devices can be too bulky or can have too small interaction surfaces for the task at hand. Data gloves are inconvenient in hot weather and for fine tool manipulation tasks. Speech recognition fails in noisy environments and is not always socially acceptable. Additionally, their use is limited for people with speech impediments. Free-hand gesture interfaces undoubtedly have their shortcomings as well, but they should be available as an interaction modality whenever desired. Hands and hand gestures are in many ways well suited to human-computer interaction (HCI) tasks. Hands are man's most dextrous physical tools as pianos and laparoscopy readily demonstrate. Human motor skills can perform tasks with high precision and incredible speeds. In the mobile context, where data input is a challenge and methods that require physical devices are at a disadvantage, vision-based hand gesture recognition has the potential to become an important interface modality.

Figure 1: Our mobile user interface in action. All hardware components aside from the display and camera are in the backpack.

In this paper, we take a systematic approach at investigating the capabilities of hand gesture VBIs capable of supporting the mobile user. In section 3, we distinguish different types of gestures and their place in the mobile user interface (MUI). We then briefly evaluate the suitability of computer vision methods to the task of hand detection, tracking, and posture recognition. In conclusion, we describe in sections 4 and 5 a system that we have built to demonstrate the feasibility of our approach in serving as the sole interface modality to a set of mobile applications. Our hardware setup, shown in Figure 1 and detailed in subsection 5.1, consists of a head-worn display with an attached camera as the only visible and interacting components, while a laptop is stowed away in a backpack. Overall, our results offer guidelines for MUI designers desiring to employ CV for gesture recognition interfaces. We hope to stimulate increased research and interest in using CV in the mobile systems arena.

2. Related Work

This section covers research in VBIs for mobile computers and applications. Work solely related to computer vision issues will be selectively addressed in our algorithm description in section 4.

Starner et al. pioneered mobile VBIs for American Sign Language recognition [32]. A cap-mounted camera found and tracked skin-colored blobs (regions) in the video and analyzed their spatial progression over time. Words were extracted from the fairly coarse trajectories with Hidden Markov Models (HMM). While their system works with non-instrumented hands just as ours, our system integrates multiple modalities (skin color and texture information) at all stages of processing to overcome the robustness limits associated with relying on the accuracy of a single-cue image segmentation. They recognized the need for a second modality and experiment now with accelerometers attached to the signer's wrists [2]. Recognizing a set of communicative gestures (that frequently exhibit distinct spatial trajectories, see Quek [24] for a classification) requires more semantic post-processing, while manipulative and discrete gestures as recognized by our methods are more demanding on the computer vision methods.

In a later project, Krum et al. built a mobile system for recognizing gestures and speech [18]. It employed specialized imaging hardware with active infrared illumination and provided a small interactive area at sternum height in front of the wearer's body. All our vision components are passive, in particular we do not require power-hungry active light sources. A related user study [17] found that the relatively static hand position for extended periods of time caused fatigue and the unwillingness for more extensive hand motion. Through use of a much larger area for potential interaction and a diverse range of gestures, fatigue and discomfort symptoms were not observed during short-term (15 minutes) informal experiments. Our system also includes a head-worn display (HMD) that allows for registered manipulation technique (see section 3).

Kurata et al.'s HandMouse [19] is a VBI for mobile users wearing an HMD and camera just as in our scenario. It differs in the fact that the hand has to be the visually prominent object in the small field-of-view (FOV) camera image and that it relies solely on the skin color as image cue modality. The robustness gained with our multi-modal approach makes it possible for the image of the hand to be much smaller, to have arbitrarily-colored backgrounds, and even to take the system outdoors. Going beyond the interaction methods they demonstrated, we characterize additional techniques and their suitability for mobility and the outdoors. Our system then shows how this improves MUI usability and effectiveness.

Dominguez, Keaton et al. [4, 14] implemented a compelling wearable VBI that enabled the user to circle objects in view with a pointing gesture. Color is the only extracted image information. Our system uses some of the same interaction concepts (registered manipulation), but provides increased robustness and mobility.

A few research project have lately come about that use the ARtoolkit [13] software to obtain the hand's 6 degree-of-freedom (DOF) position. For example, Thomas and Piekarski [33] attached a marker fiducial to the back of the hand whose position and orientation is obtained from grey-level image processing. This method's constraints are limited permissible rotations and fairly big hand sizes. We strived for unadorned hands without any markers while still capturing more than location information. They introduce another device (a Pinch Glove) for hand configuration recognition, our system instead consists entirely of computer vision methods. As a side note, the Outdoor Tinmith Backpack Computer is an extreme example of a high-fidelity wearable computer, but also of the amount of equipment required to facilitate this functionality. Our system on the other hand was designed to minimize extraneous hardware requirements and instead make the computer disappear as much as possible. Only the head-worn devices are exposed, everything else is carried in a small backpack.

Wearable Augmented Reality systems such as [6] are related in that they are a prime recipient for our interaction methods, as they lack sufficient interface capabilities and are thus still limited in their application. We have so far restricted our work to 2D interaction, but a great benefit of hand gestures are naturally their inherent 3D capabilities (see [1] for 3D UI techniques).

Not built for mobility nor for free-hand gestures, but still vividly demonstrating the realm of capabilities that can be explored with gestural commands is an interface by Wilson and Shafer. The XWand [37] constitutes a very natural gesture interface with the aid of a hand-held device that can be pointed at objects, swung, and gestured with in other ways. Also tangentially related is DyPERS [11] as it gives a

glimpse at the possibilities that permanently worn display-camera-combinations offer: Video memories are captured on demand and then automatically attached to real-world objects as they are encountered later. Memorizing people's names is just one application.

3. Hand Gesture Interaction Techniques

The hand can perform different types of actions. For the sake of clarity, we briefly classify them by their physical characteristics. A *posture* is a static configuration of the fingers and the hand. *Gesture* is a more general term as it can involve dynamic aspects of movement. Other classifications of hand gestures usually consider their meaning and interpretation (see for example [15, 22, 24]). *Static postures* are view-dependent configurations of the hand that are recognized immediately, on a per-frame basis. A configuration is described by the joint angles of all hand-related joints, but a qualitative statement suffices for most occasions. *Location-dynamic gestures* are static postures combined with a hand-global movement relative to a camera. *Orientation-dynamic gestures* involve a static posture that, over time, is rotated with respect to the observer, resulting in different views. *Ego-dynamic gestures* are configuration changes such as making a fist and then transitioning to a pointing gesture by extending the index finger. While much progress is being made to recognize and model these gestures from video (for example [20]), they are currently still too complex for reliable recognition in unconstrained environments.

Our computer vision methods target the first three types, but are able to deal with the fourth kind as well. For example, after detection of a static posture the system tracks the hand during location-dynamic gestures, continuously estimates the hand rotation, and does not lose track even despite abrupt configuration changes.

Given this physical characterization, one can distinguish three main types of gesture interpretation for UI purposes. Their characteristics and manipulation techniques that they support are described in the following paragraphs.

Pointer-based manipulation describes gestures and their interpretation as if the hand or a particular finger acts as a computer mouse–by movements in an *input plane* controlling a pointer with a linear transformation on a distinct *manipulation plane*. The input plane is fixed relative to the camera coordinate system, while the ("direct") manipulation plane is fixed relative to the screen coordinate system. This type of a gesture is unsuitable to interfaces that must be operated while in motion. Usually, a separate gesture of the location-independent kind is used for making a selection. An example in our application is the button selection. Instead of requiring an absolute match between the location of the hand and the button, only relative motions from a starting point are considered. This avoids issues of fatigue that stem from too restricted postures. This method works well for selection from a menu list, as the pointer movement can be constrained in one dimension, which reduces the required precision of the hand movement and thus likely also the associated fatigue. We found this very convenient, especially if the linear transformation involves a scaling factor that avoids large hand movements, but also is sufficiently robust to involuntary jitter during general body motion. A sub-classification can take the orientation- and ego-dynamics of the hand into account, distinguishing pointer-based interfaces dependent on hand dynamics during the hand-global motion. Examples of VBIs that allow for pointer-based manipulation can be found in Fukumoto et al. [7], Hu et al. [9], and Quek and Mysliwiec [25].

Registered manipulation is similar to pointer-based manipulation, but the input and manipulation plane are coincident. That is, the hand or finger virtually touches the object it is interacting with in a mixed reality world. Due to the difficulty of good registration in optical see-through displays, this method will currently work better in video see-through display methods. Again, it is hard to perform this kind of manipulation while on the move. This method is especially suitable to interaction with virtual objects in mixed reality scenarios and was, for example, employed in designating an area of the FOV to be captured as a snapshot within our application. Another reason for using registered manipulation in our case is that for larger active interaction regions (such as in the number selector interface) it was difficult to work around the lack of the "picking up and repositioning the mouse" action. Previously, the tracked hand would frequently exit the camera's FOV, while the pointer had not reached the desired target yet. In addition to that, interaction in registered manipulation interfaces allows for direct reaching for the target without any mouse-to-pointer translation. In other words, it is probably hard to define a linear transformation function that relates the input with the manipulation plane in a more intuitive and effective way than the identity transformation. Note that during motion, just like in the previous case, other hand configurations can be assumed.

Location-independent interaction refers to hand postures that can be performed anywhere within the camera FOV where it produces a single event. Performing predefined actions, such as generic task switching, confirmation or rejection gestures, clicking as in our application are perfect for this mode, as opposed to those that require interaction with dynamic UI elements. As Hauptman pointed out [8], pointer-based manipulation should not be the only mode of interaction. Instead, gestures are preferably performed with more than one finger and both hands. Location-independent gestures are thus an important mode of interaction, especially for people "on the move".

A selecting gesture (a "mouse click") is a necessary con-

cept for many pointer-based and registered interfaces. It can be implemented with two techniques: **Selection by action**, which involves a distinct gesture or posture that is performed to signal the desire to select. If the same hand is employed for both pointing and selection, some movement during the selection action must be expected and should not interfere with pointing precision. For high precision demands a **selection by suspension** technique might be more appropriate, in which the desire to select is conveyed by not moving the pointer for a threshold period of time. Requiring the user to be idle for a few seconds, or constantly move her hand to avoid selection, is usually unwise, particularly in mobile contexts. We therefore only used selection by suspension for the area snapshot task, for which the user will most likely assume a stationary body position and an additional gesture definition seemed more complicated. Selection by action was used for all other selection interactions.

The gesture interface was built with several constraints in mind. First, there are human factors that limit reach. In accordance with ref. [16], the interaction range was designed to be within the users' comfort zone. The comfort of hand postures (in the sense of configurations) has not yet been evaluated. Second, the gestures were selected for being natural and intuitive to the user. This is best illustrated by the pointing gesture. However, there is very little research available to this end and we have to rely on our intuition to some extent. Third, we picked gestures that were sufficiently distinguishable from background artifacts. That for example rules out using a fist from dorsal view. Gestures have to be different enough from one another that posture classification is feasible in a robust manner. Owing to the learning-based recognition method, this turned out not to be a factor. Table 3 summarizes which hand gesture interaction techniques we used for which application functionality, which will be described in detail in section 5.

manipulation technique	**prototype application's UI component**
pointer-based	voice recorder, image/video capture menu
registered	area image capture, number selection
location-independent	task switch, work order selection
selection by action	all button clicks
selection by suspension	area image capture

Table 1: The different types of gestures and which application part implemented this interaction technique.

Hauptman [8] made another important observation: The importance of immediate feedback for the user's actions. We provide immediate feedback about the system state to the user. After a few experiments we removed low-level output from the CV methods from the display (such as segmentation or tracking confidence information). Instead, we found it sufficient to only relay the most important vision-level information in a timely and direct manner: whether detection and tracking of the hand was successful. This is simply indicated after detection by a big red dot on top of what the system thinks is the hand. All other feedback comes from application space. For example, a red border is drawn around buttons that the user hovers over, signaling that executing a selection gesture will "click" that button. An iconic hand is drawn as cursor for the pointer-based manipulation techniques.

Raising the level of feedback and hiding the interface implementation details is a desirable step as it allows the user to concentrate on the task rather than the means to execute it.

4. Software Architecture – Computer Vision

The core of this paper's contribution – demonstrating feasibility of and choices for mobile VBIs – is based on the computer vision system. We use a combination of recently developed methods with novel algorithms to achieve real-time performance and robustness. A careful orchestration and automatic parameterization is largely responsible for the high speed performance while multi-modal cue integration improves robustness to achieve a low false positive rate.

There are three stages of processing. In the first stage, the presence of the hand in a particular posture has to be detected. It is undesirable to have the vision interface in an always-active state since coincidental gestures may be interpreted as commands. In addition, vision processing can be made faster and more robust if only one gesture is to be detected. After gesture-based activation of the vision interface, the second stage is entered. It serves as an initialization to the third stage and is repeatedly executed if the posture classification in the third stage succeeded. Together, the second and third stage track the hand continuously in the full-size video. Also, at the tacked location key postures are identified.

This multi-stage approach makes it possible to take advantage of less general situations at each stage. Thus, faster processing speed can be achieved by exploiting spatial and other constraints that limit the dimensionality and/or extent of the search space. We use this at a number of places in our algorithms: The generic skin color model is adapted to the specifics of the observed user (see subsection 4.2), and the search window for the hand location is predicted by Condensation (see subsection 4.3). However, staged systems are more prone to error propagation and failures at each stage. To avoid these, every stage makes conservative estimations and uses multiple image cues (grey-level texture and local color information) to increase confidence in the results.

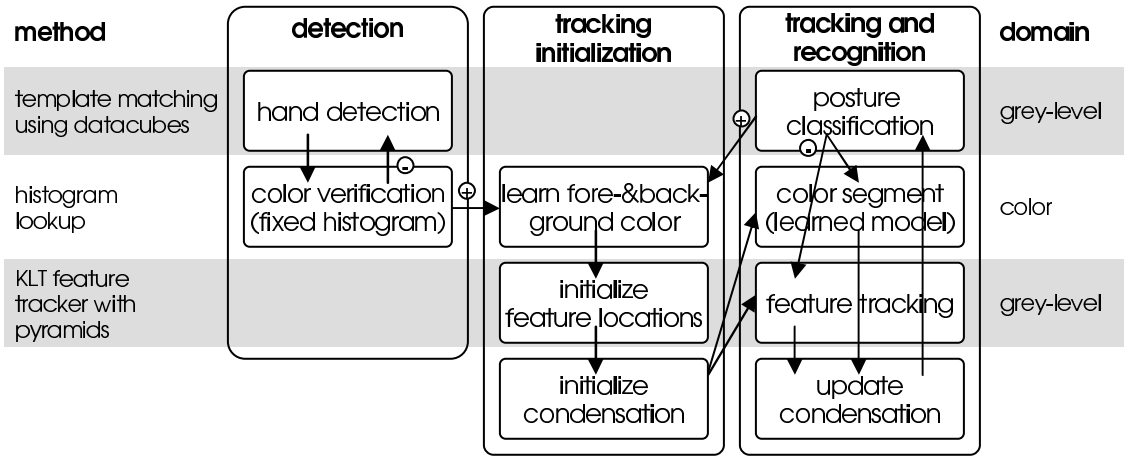

Figure 2: How the computer vision methods are arranged: Only on successful hand detection will the tracking method start operating. As long as this method has not lost track of the hand, the posture recognition is attempted. If successful, features and color are re-initialized if the condensation tracker reports low confidence.

The final output of the vision system consists of the location and sometimes the posture of the hand, and at some occasions also the location of the second hand. The location, estimated as a measure of individual features' locations, is in 2D image coordinates. The posture is described as a classification into a set of predefined, recognizable hand configurations. The diagram in Figure 2 details the components of our vision system and their interactions. The following subsections describe the computer vision methods in more detail.

4.1. Hand Detection

Most earlier gesture recognition systems place restrictions on the environment, such as a uniform background [29, 26], a static background [23], or colored gloves or markers (fiducials) on the hand [5, 33]. Hand detection against arbitrary background was achieved for example by Triesch and Malsburg [34] who are able to distinguish hand poses with 86.2% accuracy, or with a method by Cui and Weng [3]. Neither of them however can perform the detection in real time as is required for UIs.

We customized an object detection method recently proposed by Viola and Jones [36]. Objects are learned during a training phase with Ada boosting of features that compare grey-level intensity in rectangular image areas. The decisive advantage of this over other appearance-based methods is that it can be implemented with "integration templates", a method borrowed from database research and better known as *data cubes*. During detection, a pre-computation step produces a 2-dimensional brightness integral. The sum of pixel values in arbitrary rectangular areas can then be computed in constant time. Training of the approximately 200-feature classifiers (one classifier for each posture and view) on 200 test and 200 validation images took about a day on a 25-node PC cluster running Linux. In contrast, *detection* of hands of arbitrary scale (larger than 30x20 pixels) is very fast and can run with about 10fps on a 640x480 sized video stream on a high-performance laptop.

There are other accurate methods that compress texture features into a low dimensional space, for example the principal component analysis (PCA) in Eigenfaces [35], or Gabor wavelets in [28]. For every area that is to be classified they need to inspect every pixel however, which prohibits their performance to reach the demands of interactive systems.

The initial hand pose is a top-down view of the flat hand with the fingers touching each other (see Fig. 4). We chose this posture/view combination due to its highly identifiable nature against background noise and therefore its good success rate as a fail-safe detection condition. The recognition is executed in a part of the camera's FOV that corresponds to a natural reaching distance in front of the right shoulder. The original object detection method is very sensitive towards in-plane rotations. We trained one detector each for multiple slight rotations of the same hand posture (four incremental rotations of 5° each), allowing the initial posture to be performed at angles convenient to the user. The same technique was used for the classified postures as well, but with fewer angles.

Upon detection of a hand area, it is tested for the amount of skin colored pixels it contains. To this end, we built a histogram-based statistical model in HSV space from a large collection of hand-segmented pictures from many imaging sources, similar to Jones and Rehg's approach [12]. Color-based methods bank on the fact that the appearance of skin color varies mostly in intensity while the chrominance remains fairly consistent [27]. We used a histogram-based method because they achieve better results in general, user-independent cases. If a sufficient amount of area pixels are

classified as skin pixels, the hand detection is considered successful and control is passed to the tracking initialization stage.

4.2. Tracking Initialization

The very general statistical model of skin color is then refined by learning the observed hand color on the area detected as a grey-level texture. This color histogram is contrasted to a reference area that is assumed to not contain skin areas, located around the hand area to the left, top and right. This assumption always held in our experiments due to the camera FOV and angle. Note that even though other skin-like colored objects might be in this reference area they are still to be considered background and the learned color classifier automatically incorporates this information by modeling the observed foreground/background color distribution. Figure 3 exemplifies the color segmentation and feature tracking operation.

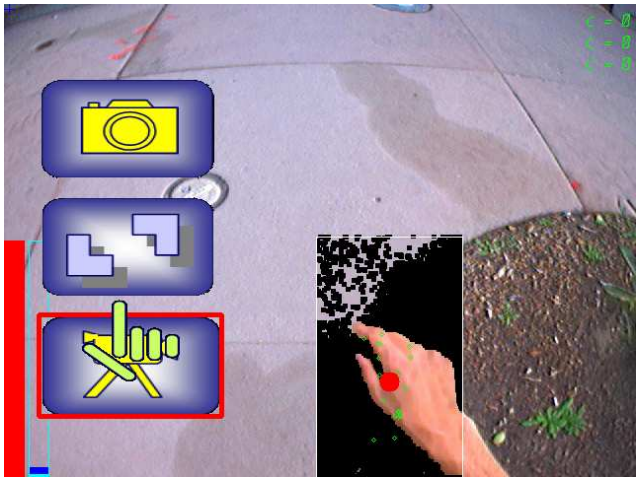

Figure 3: This is a screen capture with CV-debugging output turned on. Visible is the partially color-segmented video frame, illustrating how skin color by itself is not a reliable modality. Also shown are the KLT features at their current location.

Next, about twenty KLT features [31] are placed randomly on skin-colored spots in the detected area. KLT trackers are named after Kanade, Lucas, and Tomasi who found that a steep brightness gradient along at least two directions makes for a promising feature candidate to be tracked over time. In combination with image pyramids (a series of progressively smaller resolution interpolations of the original image), a feature's image area can be matched efficiently to a similar area in the following video frame (see [21]).

KLT features do not encode object-level knowledge nor global information. To achieve consistency among the features, to improve tracking across changing backgrounds, and to better deal with short occlusions, we enforce global constraints on the features' locations with the aid of Condensation tracking [10]. This is a particle filtering method that models very well the multi-modal probability distributions that arise during object tracking due to multi-causal background noise. The modeled state in our case consists of the 2-dimensional location of the centroid of the features and an angular component that describes the rotation of all features around that centroid. During the initialization step, this state is set to the current position without rotation, and sensible (spatial and angular) velocity bounds are set, derived dynamically from the match area size and offline experimentation.

4.3. Tracking and Recognition

On receiving another input frame, the locations of the features are updated and the posture at the tracked hand position is classified. First, a search area is determined by the previous tracked location and hand velocity. With help of the learned foreground/background color model, the pixels within this search area are evaluated for their probabilities to be of skin color. Next, the KLT features' positions are moved to the location predicted by the Condensation tracker. From there, they are updated with the traditional pyramid-based feature matching algorithm. This and the color probability map constitute the observation or measurement for the Condensation update step: many randomly generated state samples are confidence-rated based on their distance to the observation. This distance is additionally weighted by the summative color probability of the image patch at the feature location. This method leads to a very natural integration of feature movement based on grey-level image texture with texture-less color information. In addition, it enforces global constraints on the feature locations, keeping outliers at bay and following the main object of interest instead.

There is one aspect that must be considered. The method as described would work fine for rigid objects with mostly invariant appearance. However, this is not the case for hands, a highly articulate object whose appearance can change vastly and rapidly. The feature match correlation in two consecutive frames can potentially be very low at times. Other features will be moved into a low-match-correlation area due to the global constraint. To cope with this situation, our algorithm removes these features from the set and re-initializes them at positions that have a high skin color probability and are within a certain range of the centroid. Beyond that, they are chosen randomly. This technique is responsible for the system's ability to not loose tracking despite sudden or vast ego-dynamic gestures (posture changes).

We found edge and shape based methods unsuitable for detection and tracking in complex environments. Their robustness is largely determined by the amount of contrast between the foreground object and the background scene,

which can not be guaranteed. Also, the frequently associated gradient-descent methods that enforce global constraints for the individual edges' locations encounter problems with the highly articulated hands whose appearance can trap the algorithms in deep local minima because of the multitude of strong edges. Also, texture-based features, which have true spatial extent, contain more information than line features. Thus, edge-based methods need further processing to reach the same level of knowledge integration. This level dependency makes them more susceptible to image noise. We also decided not to go beyond appearance-based methods to extract hand configuration information, such as with kinematic 3D hand models, because the available methods do not exhibit the performance characteristics (robustness and accuracy) necessary for VBIs due to too many degrees of freedom and arising singularities during parameter estimation.

The described combination of feature matching, color segmentation, and Condensation tracking is by itself a good object-independent tracker. Over time however, the only guarantee we get about the tracked object is that it has a high content of skin color. Nothing prevents the features from moving onto other skin-colored objects. To counter this drift, the tracker attempts detection of a set of key postures at every frame with the detection method described in subsection 4.1. We have currently only trained it for five postures (see Fig. 4) and a number of in-plane rotations, which proved to be sufficient to re-initialize tracking frequently enough during gesture transitions. Once any posture is recognized, the color model is re-learned from the area known to be of skin color. Also, all features are located onto image parts that are known to belong to the hand.

Figure 4: Two examples each of the five hand postures that are recognized along with in-plane rotations within approximately 30°, and additionally around 90° for the pointing and the rightmost posture. They are shown in the minimum resolution required for recognition, 25x25 pixels, some with a distorted aspect ratio for recognition performance reasons. Currently we detect and recognize only gestures of the right hand.

4.4. Performance

The quality and usability of an image-based user interface is determined by three main aspects of the computer vision method(s) used: speed, accuracy, and robustness. Speed concerns the system's latency and frame rate. A maximum latency between event occurrence and system response of 45ms has been found to be experienced as "no delay" by Sheridan and Ferrell [30]. While we have not quite achieved that end-to-end latency, all methods combined typically require less than 100ms total processing time per frame (latency from frame capture to render completion time as reported by DirectShow on a high-performance laptop). This is well below the threshold of 300ms for when interfaces start to feel sluggish, might provoke oscillations, and cause the "move and wait" symptom [30]. It certainly was sufficient to comfortably control our prototypical applications. The system achieved frame rates of 10-15Hz. In comparison to other mobile VBIs, our method is significantly more responsive than the Hand Mouse [19], judging from a video available from their web site.

The accuracy of the methods—that is, the correlation of the actual with the recognized hand gestures—determines largely the degree of user satisfaction. The object detection and posture recognition methods were trained to have a very low false positive rate (smaller than 1e-10) and a medium detection rate between 85% and 95%. In practice, and in combination with the color cue, we had not one occurrence of a false detection match and a couple of posture misclassifications per hour. A formal evaluation is in preparation. The accuracy of the centroid KLT feature's location (which we used as the input pointer's location) with respect to some fixpoint on the hand can not be guaranteed because of the entirely object-independent tracking method. This was only of concern for the registered manipulation tasks, as the other interaction techniques involve pointer location transformations or are location independent. The precision of this feature's motion was very good, even minute movements of the hand were tracked and evaluated with ease.

Robustness is the method's ability to deal with different environmental conditions, including different lighting (fluorescent and incandescent lighting, sunlight), different users, cluttered backgrounds, occlusions (by other objects or self-occlusion), and non-trivial motion. Using multi-cue vision methods, in our case pure grey-level information in combination with color validation and segmentation, can overcome some of the problems associated with the uncertainties of unconstrained environments. In particular, our methods are designed be invariant to the automatic image quality adjustments of digital cameras, such as white balance, exposure, and gain control. Three conditions will still violate our assumptions and might impact recognition and tracking negatively: An extremely over- or under-exposed hand appearance does not contain a sufficient amount of skin-colored pixels for successful detection, based on the fixed HSV histogram-based segmentation. Second, if the color changes dramatically in between two consecutive successful posture classifications, the tracking degenerates into

single-cue grey-level KLT tracking. Since the system updates its color model periodically, it is able to cope with slowly changing lighting conditions, however. Third, shadows cast onto the hand change the appearance dramatically and can currently not be handled during detection and they also impact tracking.

None of the system's functionality explicitly detects or models hand occlusions. However, brief occlusions of the tracked hand with foreign objects or the other hand do generally not cause all KLT features to be lost. Thus, unless very prominent skin-colored gradients exist on nearby objects, the Condensation-based global constraint imposed by the tracker has a good chance of settling the features back on the hand once the occlusion has passed. Head (and thus camera-) motion with a motionless hand can currently not be distinguished from the inverse hand motion. In this case, feature tracking is in fact more reliable because the whole scene including foreground and background merely undergoes an apparent translation, as opposed to non-linear transformations caused by objects moving relative to each other.

The implemented CV methods are largely camera-independent: The detection and posture recognition classifiers were trained with images taken with different still picture cameras, while the system was successfully tested with three different digital video cameras. In addition, none of the training images was shot with as short a focal length lens as our mobile camera has. These facts suggest that the entire system will run with almost any color camera available.

5. The Wearable Computer

5.1. Hardware Setup

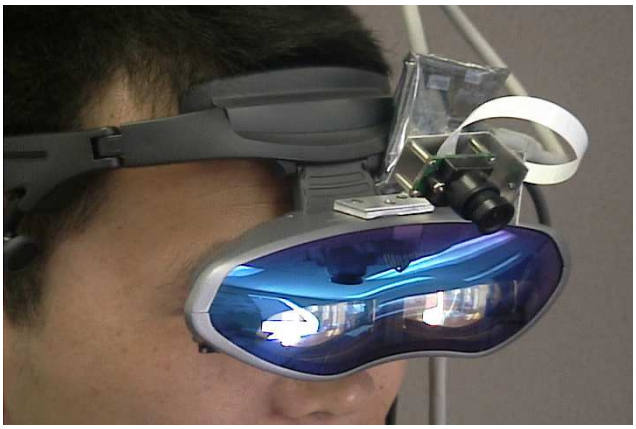

Figure 5: A user wearing the display and camera. All other devices are stowed in a conventional backpack and need not be accessed.

The hardware setup of our system produces output through a head-worn display (HMD, Sony Glasstron LDI-A55), atop which we mounted a small digital camera ("FireFly", Point Grey Research), see Fig. 5. The camera has a horizontal FOV of 70° and its pitch for a normal head position is adjusted to cover the range from almost straight down to horizontal. The live video stream is fed into the display to achieve an optical see-through effect. This also alleviates problems with the HMD's small 30° FOV because it make 70° FOV available to the wearer. The resulting spatial compression takes users a few minutes to get used to, partially because of the scale change of visual feedback, but no adaptation problems were reported after that time. Use of this fisheye-style lens reduced the tunnel effect that most video see-through mixed reality displays exhibit.

The high FOV is important for interface functionality as well because the hands are visible along with the a more forward-facing view direction. It allows interaction methods where the input plane is co-located with the interaction plane, as opposed to decoupled as with the mouse and computer screen.

A laptop computer, stored along with batteries and a few adapters in a conventional backpack, receives the video input, processes it as detailed in section 4, adds the output information from the three applications described in the following subsection, and sends the result to the display as an NTSC signal. Note that no other input device such as a Twiddler keyboard or 3D mouse are used, instead, the interface is combined into a single head-worn unit. Other than more traditional wearable computers, our system only consists of these three parts, laptop, HMD, and camera, making it a fairly easy to assemble and relatively inexpensive mobile computer.

5.2. User Application

The application we used to test the functionality of our VBI was custom built for this purpose and designed to demonstrate the suitability of VBIs for the mobile use. We thus made certain assumptions about the area and situation of deployment, which will be described as well. No multimodal interface capability was provided as we wanted to focus on the computer vision aspect, however the addition of a speech recognition component is likely to be beneficial for many aspects of usability and flexibility.

We composed a set of applications that supports building facilities managers in their daily tasks of performing maintenance operations and immediate-attention work requests, for example investigating a water leak or power failure in a particular room. The wearer of our mobile system can utilize three main applications: an audio recorder, a digital still and video camera, and a work order and communication application. These applications run continuously, however only one of them is in the foreground at any time. The foreground application is selected by performing a task-switch gesture for a short period of time, which cycles through the applications and a "blank screen" mode, one by one. The applications have the capability to display status informa-

tion and/or alerts even when running in the background, such as the voice recorder showing an icon while recording sound, and the communication application displaying alerts for incoming messages. User input, however, is only possible in the foreground application.

5.2.1 Voice Recorder

Figure 6: The voice recorder interface in the foreground. Note that the interface snapshots shown in the various figures were taken in different environments, illustrating the ability of our system to adjust to varying backgrounds and lighting conditions. This snapshot was taken while walking.

A small microphone clipped to the shirt allows auditory recordings, activated by gesture commands that start, pause, resume, and stop a sound recording. It is then given a unique numerical identifier which is used to attach it to a work order reply. We currently do not provide the possibility to review the recordings and play them back because this is a straight-forward extension that does not demonstrate additional interface properties. This interface was designed with the pointer-based manipulation technique and consists of three buttons. Naturally, they are selected by moving the hand pointer (in any posture) over the button and performing the "select" posture. We chose pointer-based manipulation for three reasons. First, it allows us to snap the pointer to the default button, so that in most cases only the selection gesture has to be performed. While this behavior might be disruptive in a desktop environment, it is a lot more convenient within the MUI context. The snap functionality amounts to a dynamically determined, but then constant offset between the input- and manipulation planes. Second, we placed a restriction on the pointer movement to the horizontal dimension, making the pointing task a linear left-right motion operation. This makes the interaction much easier than true 2D selection, especially while walking. Third, hand movements are translated into proportionally larger pointer movements to allow for big, easily visible buttons while not demanding equally extensive hand movements. This scale factor was determined empirically. Whenever the hand pointer is in the area of a button (hovering above it), it is shown with a red border surrounding it. This was implemented to give the user a stronger visual sensation for what would happen if s/he "selected" at the current location. While recording sound, the symbol of a running tape recorder is shown in the upper area of the screen. When the recording is stopped, the recording's identifier is prominently displayed for a short time period. Figure 6 shows the voice recorder interface in the foreground.

5.2.2 Image and Video Capture

The image capture application has three modes of operation which are selected via buttons similar to the voice recorder interface, shown when the application is brought into the foreground and not actively recording. The interaction technique with the image/video capture menu is very similar to that of the voice recorder, only that the buttons are arranged in a vertical fashion and the pointer movement is constrained to that dimension. The first mode allows a user to take a single image of the entire visible area. A count-down timer is overlaid after activating this mode. A picture is taken and stored at the end of the count-down, along with an auditory confirmation. All pictures and video recordings are also given unique identifiers, briefly displayed at the end of the capture. We found that users usually decide on a view and settle their motion after around five seconds, so that is what we chose as the time delay. This mode is the easiest to use, but does not introduce a new interaction capability over the voice recorder interface.

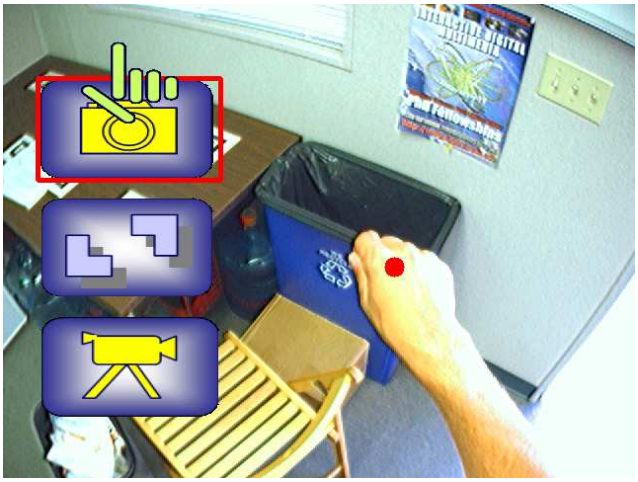

Figure 7: This view shows the image and video capture application's main menu.

The second mode enables snapshots of selective areas within the camera's FOV, which demands a way to select multiple points in the image at the same time. (Implementations of the same functionality that use only one pointer

are feasible but less convenient to use.) After selecting this mode, the computer vision methods search for the left hand as the nearest distinct skin-colored blob to the left of the (already tracked) right hand. Once found, the rectangular area enclosed by both hands is highlighted in the display. Once the positions of both hands stabilized with respect to the camera, a picture of the selected area is taken and stored. It is also displayed to the user for a few seconds, along with its identifier. Figure 8 shows the highlighted area inside the region selected by the user's hands.

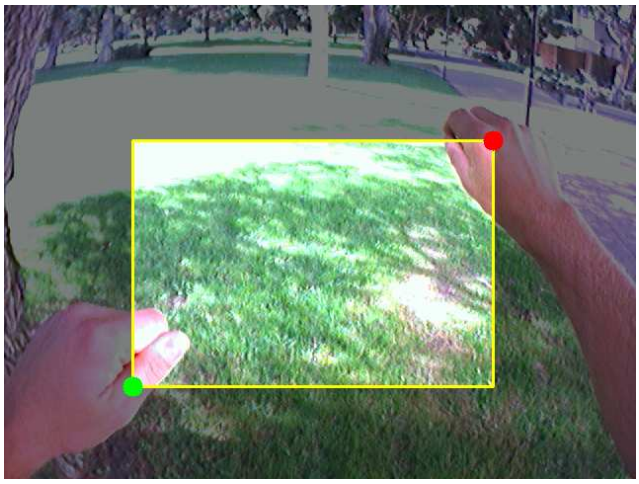

Figure 8: Taking a picture with the image recorder application. The user has selected the area that is to be recorded, and the image will be captured if the hands have settled for five seconds.

The third mode of operation of the image capture application records a continuous video stream of the camera's entire FOV. It is started with five seconds delay after this mode is selected, again with a count-down timer indicating the time delay. The only interaction functionality after the video starts is to stop the recording, which occurs as soon as the hand is detected within the interaction initiation area (within the camera's FOV).

5.2.3 Work Order Scheduler

To complete the functionality set for the test scenario for the vision-based wearable system, a scheduler was added. With the aid of this application, the person in the field is enabled to retrieve, view, and reply to work requests. Continuous connectivity was not a requirement, rather, the application was designed similar to an intermittently connected mail client that automatically switches from offline to online mode and back, depending on network resource availability.

As this application was brought into the foreground, up to three work orders are shown at a time as horizontal bars. Indicated on them are a title and the status of the order (open, closed, follow-up). Attachments (pictures, video, sound recordings) are shown as a combination of their identifiers and type. If there are more work requests than the screen can hold, the items can be scrolled up and down as indicated by a special scroll tap area at the top and bottom of the listed work orders (see fig. 9). Three dedicated, static hand gestures allow for selection and manipulation of work requests: One gesture selects the work order above the current one, another gesture selects the one below the current one. Scrolling happens automatically if the top or end of the visible part of the list was reached. We choose the discrete posture technique over pointer-based manipulation because scrolling with pointers is an awkward operation as anybody who had to scroll while dragging-and-dropping will confirm. The third gesture facilitates activation of the currently selected work order. This brings into the foreground the work order editor. It displays the entire text of the work request, which in the presented version consists of unformatted text. Repetition of the gesture used for entering the editor also signals to change the status of the work request. A subsequent gesture decides whether an attachment is to be added or not. Attachments can be selected from the previously recorded media clips (voice recording, still picture, or video). The selection process is realized with registered hand movements. This was decided based upon the fact that there could be a large set of attachments to choose from which can be randomly selected. This would make access in a sequential fashion such as with the work order selector very inconvenient. The selection gesture picks the currently highlighted number. The attachment process can be repeated until all desired attachments are appended to the work order reply. As the last step, the work order status can be changed into *done* or *follow-up required*. This choice is

Figure 9: Scrolling through the list of work order summaries. Note that the hand location (big red dot) does not fall on any of the interaction surfaces. Instead, the currently pre-selected item is chosen by discrete postures ("up"/"down" gestures).

implemented using the same discrete gestures that the work order selection is operated with.

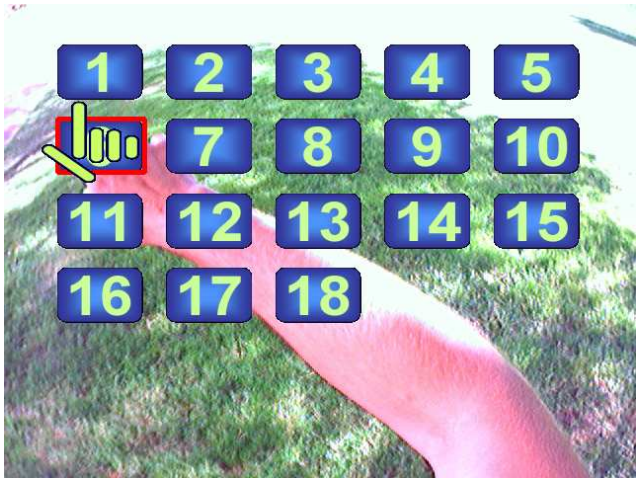

Figure 10: The number selector is operated with registered manipulation. Moving the hand over one of the numbers and performing a selection gesture picks the respective attachment.

6. Future Work

We have not found a good solution to automatic detection of tracking loss. Heuristics based on KLT feature locations provide some clues, but in a few occasions the system would track some non-hand object. This left the user no other chance but "catching" the features on the erroneously-tracked object. We have also not yet implemented a specific gesture to purposefully end active tracking. Right now one has to move the hand out of the FOV of the camera so that the tracking algorithm detects failure and the system re-initializes.

Depth (the distance from the camera) and world-referenced 3D information can supply important additional input parameters, especially for manipulation of virtual 3D objects. For this to work, the accuracy of the hand location has to be improved as well (see subsection 4.4). These are the main areas that we would like to extend our system into. So far, we experimented only with limited two-handed manipulation, but would like to explore that interaction technique more. Recognizing dynamic gestures such as clapping for interface purposes is also on the horizon of interesting system extensions.

A formal evaluation of the performance of the CV methods for gesture detection, tracking, and recognition is in the making. No multi-modal interface capability is currently provided because we wanted to focus on the computer vision aspect for this paper. However, the addition of a speech recognition component is certainly going to be beneficial for many aspects of usability and flexibility.

7. Conclusions

We showed different manipulation techniques for hand gesture-based mobile user interfaces and presented computer vision methods capable of detecting these gestures. We presented a prototype system that we built to demonstrate robustness and usability of this vision-based interface when used as the sole input modality to a wearable computer. Robustness was evidenced with regard to environment conditions, in particular to indoor and outdoor lighting, cluttered backgrounds, concurrent movement of camera and user, user independence, and camera independence. Good usability results from relatively low interaction latencies and the aforementioned robustness.

The contribution of the presented work is twofold: Firstly, the system shows the feasibility of vision-based hand gesture interfaces as the exclusive input modality for wearables. Secondly, it demonstrates enhanced interaction capabilities through hand gesture recognition, some of which are difficult to achieve with other modalities. We conclude that computer vision has reached a stage where it can effectively replace some physical-device interfaces and augment others, greatly enhancing the usability experience of the mobile user.

References

[1] D. A. Bowman, J. J. L. Jr., M. R. Mine, and I. Poupyrev, *Advanced topics in 3d user interface design*, August 2001.

[2] H. Brashear, T. Starner, P. Lukowicz, and H. Junker, *Using Multiple Sensors for Mobile Sign Language Recognition*, in IEEE Intl. Symposium on Wearable Computers (ISWC), Oct 2003.

[3] Y. Cui and J. Weng, *A Learning-Based Prediction and Verification Segmentation Scheme for Hand Sign Image Sequence*, Transactions on Pattern Analysis and Machine Intelligence, (1999), pp. 798–804.

[4] S. M. Dominguez, T. Keaton, and A. H. Sayed, *Robust Finger Tracking for Wearable Computer Interfacing*, in ACM PUI 2001 Orlando, FL, 2001.

[5] K. Dorfmüller-Ulhaas and D. Schmalstieg, *Finger Tracking for Interaction in Augmented Environments*, in IFAR, 2001.

[6] S. Feiner, B. MacIntyre, T. Höllerer, and T. Webster, *A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment*, in Proc. First Int. Symp. on Wearable Computers, October 1997.

[7] M. Fukumoto, Y. Suenaga, and K. Mase, *Finger-Pointer: Pointing Interface by Image Processing*, Computers & Graphics, 18 (1994), pp. 633–642.

[8] A. G. Hauptmann, *Speech and Gesture for Graphic Image Manipulation*, in ACM CHI, May 1989, pp. 241–245.

[9] C. HU, L. LIANG, S. MA, AND H. LU, *Virtual Mouse–Inputting Device by Hand Gesture Tracking and Recognition*, in Proc. IEEE Intl. Conference on Multimodal Interface, October 2000.

[10] M. ISARD AND A. BLAKE, *Condensation – Conditional Density Propagation for Visual Tracking*, Int. Journal of Computer Vision, (1998).

[11] T. JEBARA, B. SCHIELE, N. OLIVER, AND A. PENTLAND, *DyPERS: Dynamic Personal Enhanced Reality System*, in Image Understanding Workshop, November 1998.

[12] M. J. JONES AND J. M. REHG, *Statistical Color Models with Application to Skin Detection*, Int. Journal of Computer Vision, 46 (2002), pp. 81–96.

[13] H. KATO AND M. BILLINGHURST, *Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System*, in Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality, October 1999, pp. 85–94.

[14] T. KEATON, S. M. DOMINGUEZ, AND A. H. SAYED, *Snap&tell: a vision-based wearable system to support web-on-the-world applications*, in Digital Image Computing – Techniques and Applications Conference, January 2002.

[15] A. KENDON, *How gestures can become like words*, Cross-cultural perspectives in nonverbal communication, (1988), pp. 131–141.

[16] M. KÖLSCH, A. BEALL, AND M. TURK, *The Postural Comfort Zone for Reaching Gestures*, in HFES Annual Meeting Notes, October 2003.

[17] D. M. KRUM, O. OMOTESO, W. RIBARSKY, T. STARNER, AND L. F. HODGES, *Evaluation of a Multimodal Interface for 3D Terrain Visualization*, in IEEE Visualization, October 27-November 1 2002, pp. 411–418.

[18] ———, *Speech and Gesture Multimodal Control of a Whole Earth 3D Visualization Environment*, in VisSym '02, Joint Eurographics – IEEE TCVG Symposium on Visualization, May 2002.

[19] T. KURATA, T. OKUMA, M. KOUROGI, AND K. SAKAUE, *The Hand Mouse: GMM Hand-color Classification and Mean Shift Tracking*, in Second Intl. Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems, July 2001.

[20] J. LIN, Y. WU, AND T. S. HUANG, *Modeling the Constraints of Human Hand Motion*, in Proceedings of the 5th Annual Federated Laboratory Symposium, 2001.

[21] B. D. LUCAS AND T. KANADE, *An Iterative Image Registration Technique with an Application to Stereo Vision*, in Proc. Imaging Understanding Workshop, 1981, pp. 121–130.

[22] D. MCNEILL, *Hand and Mind: What Gestures Reveal about Thoughts*, University of Chicago Press, 1992.

[23] T. MORRIS AND O. S. ELSHEHRY, *Hand segmentation from live video*, in The 2002 Intl. Conference on Imaging Science, Systems, and Technology, UMIST, Manchester, UK, 2002.

[24] F. K. H. QUEK, *Eyes in the Interface*, Image and Vision Computing, 13 (1995).

[25] F. K. H. QUEK, T. MYSLIWIEC, AND M. ZHAO, *FingerMouse: A Freehand Pointing Interface*, in Proc. Int'l Workshop on Automatic Face and Gesture Recognition, June 1995, pp. 372–377.

[26] J. M. REHG AND T. KANADE, *Visual Tracking of High DOF Articulated Structures: an Application to Human Hand Tracking*, in Third European Conf. on Computer Vision, May 1994, pp. 35–46.

[27] D. SAXE AND R. FOULDS, *Toward robust skin identification in video images*, in 2nd Int. Face and Gesture Recognition Conf., September 1996.

[28] R. SCHMIDT-FERIS, J. GEMMELL, K. TOYAMA, AND V. KRÜGER, *Hierarchical Wavelet Networks for Facial Feature Localization*, in Proc. of the 5th International Conference on Automatic Face and Gesture Recognition, May 2002.

[29] J. SEGEN AND S. KUMAR, *GestureVR: Vision-Based 3D Hand Interface for Spatial Interaction*, in The Sixth ACM Intl. Multimedia Conference, September 1998.

[30] T. SHERIDAN AND W. FERRELL, *Remote Manipulative Control with Transmission Delay*, IEEE Transactions on Human Factors in Electronics, 4 (1963), pp. 25–29.

[31] J. SHI AND C. TOMASI, *Good features to track*, in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Seattle, June 1994.

[32] T. E. STARNER, J. WEAVER, AND A. PENTLAND, *Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video*, IEEE Transactions on Pattern Recognition and Machine Intelligence, 20 (1998), pp. 1371–1375.

[33] B. H. THOMAS AND W. PIEKARSKI, *Glove Based User Interaction Techniques for Augmented Reality in an Outdoor Environment*, Virtual Reality: Research, Development, and Applications, 6 (2002).

[34] J. TRIESCH AND C. VON DER MALSBURG, *Robust Classification of Hand Postures against Complex Backgrounds*, in Proc. IEEE Intl. Conference on Automatic Face and Gesture Recognition, October 1996.

[35] M. TURK AND A. PENTLAND, *Eigenfaces for Recognition*, J. Cognitive Neuroscience, 3 (1991), pp. 71–86.

[36] P. VIOLA AND M. JONES, *Robust Real-time Object Detection*, Int. Journal of Computer Vision, (2002).

[37] A. WILSON AND S. SHAFER, *XWand: UI for Intelligent Spaces*, in ACM CHI, 2003.