Robust Computer Vision-Based Detection of Pinching for One and Two-Handed Gesture Input

Andrew D. Wilson
Microsoft Research
One Microsoft Way
Redmond, WA
awilson@microsoft.com

ABSTRACT
We present a computer vision technique to detect when the user brings their thumb and forefinger together (a *pinch* gesture) for close-range and relatively controlled viewing circumstances. The technique avoids complex and fragile hand tracking algorithms by detecting the hole formed when the thumb and forefinger are touching; this hole is found by simple analysis of the connected components of the background segmented against the hand. Our Thumb and Fore-Finger Interface (TAFFI) demonstrates the technique for cursor control as well as map navigation using one and two-handed interactions.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. Input devices and strategies; B 4.2 Input devices

Keywords: gesture, computer vision, bimanual interaction, navigation, hand tracking

INTRODUCTION
Computer vision is an attractive technique for implementing gesture input systems where the goal is the capture of fluid, unencumbered motion of the user's hands or body. Most research prototypes demonstrating this approach use a standard detection and tracking paradigm, wherein sophisticated pattern recognition techniques recover the position and shape of the hands, for example. Robustness and reasonable computational complexity are often difficult to achieve. Another consideration is supporting interaction states that allow the user to acquire and release the input, as well as select objects [4]. For example, a tracking technique capable of distinctions in hand shape might be used to enable gesture-based interaction only when the hand is in a particular configuration (e.g., extended index finger [11]).

We contribute a technique to detect when the user brings their thumb and forefinger together (to form a *pinch* gesture) from close-range video of the hands, without relying on complex hand tracking methods. The pinch detection technique provides a natural and fluid way to enable and disable gesture interaction, as well as indicate hand position and coarse shape. We present the basic mechanism and prototype, as well as the strengths and limitations of our approach.

UIST'06, October 15–18, 2006, Montreux, Switzerland.

Figure 1: TAFFI prototype includes a USB camera mounted on the top of the display, looking at the user's hand above the keyboard. Here TAFFI is used to navigate Virtual Earth aerial imagery.

PINCH GESTURES
The use of hand pose in gesture-based interfaces has a long history, particularly in glove input and vision-based systems [2, 7]. The pinch gesture has a number of attractive features for gesture-based input. It is evocative of grabbing or picking up an object, and so offers a natural signal to select or move an object in an interactive system. In the case of moving (dragging) while pinching, the motion evokes dragging an object, or tugging or stretching a piece of fabric.

The pinching grasp is precise and stable [1]. The hand is able to bring the thumb and forefinger together and apart quickly, and from the user's point of view there is little ambiguity whether the thumb and forefinger are touching. This distinction impacts detection processes: other poses (such as the extended index finger) are inherently ambiguous by comparison. Pinch gestures are relatively easy to detect if the sensing technology provides detailed information on the configuration of the hand, or when contacts are embedded in the fingertips of a glove (as in the Fakespace Pinch Glove [3]). Computer vision-based approaches to the detec-

tion of pinching are often based on precise hand shape analysis or fingertip tracking [10].

PINCH DETECTION

Figure 2 (top left) shows a typical image from a camera placed well above the keyboard, showing the user's hand as the thumb and forefinger are brought together. The technique is driven by the observation that from an overhead view, a large "hole" is created in the middle of the hand shape when the thumb and forefinger are brought together. The appearance of this hole can be detected by image *segmentation* and *connected components analysis*, both standard techniques in computer vision. In the following we detail the application of these algorithms.

In graph theory, there exists a path between any two vertices of a *connected component*. In image processing, such components refer to connected groups of identically labeled pixels in a binary image; often each component corresponds to a distinct object which is subsequently analyzed. In the case of Figure 2 (top right), the hand above the keyboard is a single connected component. With the thumb and forefinger together, however, the background texture consists of two components: the hole formed by the hand is a connected component distinct from the large background component.

To find any holes formed by the hand, a binary image is first produced in which each pixel corresponding to the background is 'on'. There are a variety of techniques to produce such an image segmentation; the current prototype compares each pixel value to that of a stored background image. If the pixel is significantly brighter than the corresponding background pixel it is deemed part of the hand and is labeled 'off", otherwise it is labeled 'on'. This scheme allows for parts of the background to become arbitrarily darker (due to shadows cast by the hand) without affecting the segmentation.

While flood fill techniques are still sometimes used to compute connected components from a binary image, there is an efficient algorithm that requires only a single pass of the image [6]. With this algorithm it is also possible to compute statistics for each connected component, such as pixel count and spatial moments (mean and covariance of all pixel positions that belong to the component). A second pass is required to generate a new image which labels each pixel of the input image with the component that contains it.

The largest background component usually corresponds to the background surrounding the hand, while any remaining background components of significant size usually correspond to holes formed by the hand. In practice, it is better to define hole components as those background components of significant size that have no pixels on the border, rather than relying on size alone—this strategy avoids false hole candidates formed when the hand cuts across the corners of the input image, for example, and accepts only holes that lie entirely within the image. This border check is performed easily by referring to the labeled image produced in the second pass of the connected components algorithm.

The pinch detection algorithm may be summarized as:

1. Obtain a binary segmentation of the hand and background.
2. Compute connected components of the background pixels from the binary image. Label each border pixel with the component that contains it.
3. Take components of significant size (in number of pixels) that do not have pixels on the border of the image. Each of these is a 'hole' indicating a pinching hand.

While this algorithm applies known computer vision techniques, it contributes a direct and robust means to realize pinching gestures in vision-based user interfaces.

TAFFI PROTOTYPE

While many advanced sensing-based gesture interfaces are designed for interaction away from the desktop, it is valuable to consider the application of these techniques to interaction just above a keyboard. The space above the keyboard is a natural site for gesture input, as it enables nearly effortless switching between keying and gesture interaction, and allows for precise sensing of one or both hands.

In previous work we explored the use of a camera above the keyboard to sense hand motion [13]. Users touched a capacitive touch sensor in the mouse button to enable gesture interaction. This mechanism does not allow the simultaneous use of two hands, and some users had difficulty precisely coordinating the motion of one hand and enabling and disabling the interaction with the other hand.

Our Thumb and Fore-Finger Interface (TAFFI) prototype adopts a similar configuration and demonstrates the pinch detection technique as a basis for gesture input. The pinch technique offers a simple and natural way to enable and

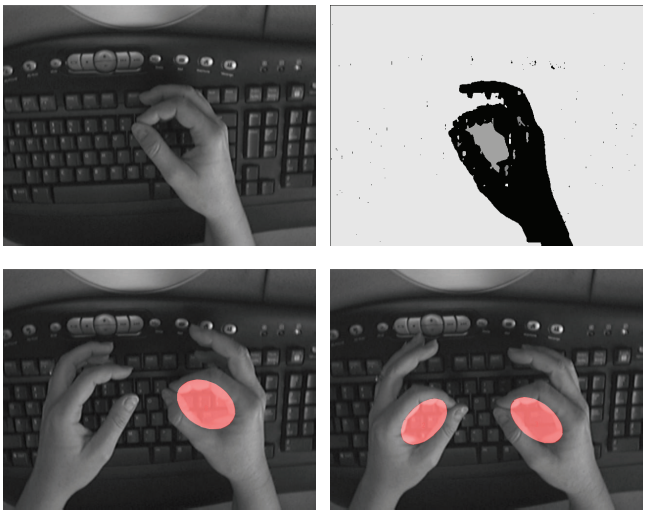

Figure 2: Upper left: Typical image of hand with thumb and forefinger together, acquired from camera above the keyboard. Upper right: Image segmentation indicates background pixels. The hole formed by the thumb and forefinger is detected as a distinct connected component (shown here darker in color). Bottom left: Ellipsoidal model of hole component for the closed hand, shown in red. Bottom right: two hands are detected.

disable gesture interaction. TAFFI also demonstrates a number of interactions beyond cursor control.

TAFFI uses a Logitech laptop USB web camera mounted on the display, which views the hands above the keyboard (see Figure 1). Full resolution (640x480) grayscale images are acquired at a rate of 30Hz. The background image required for the segmentation process described above is collected and stored when the TAFFI process is started (it is assumed that no hands are in view).

Cursor Control

The pinch detection algorithm and position (centroid) of the detected hole formed by the thumb and forefinger can be used for simple cursor control. In this mode, cursor movement is enabled only when pinching is detected. Analogous to the mouse, relative motion is computed from the current and past position of the detected hole. The pinch-to-move mechanism allows a very natural clutching behavior, much like the mouse. TAFFI uses a Kalman filter to smooth the motion of the cursor.

Many interfaces require a mechanism for object selection, and a full mouse emulation must include a way to emulate mouse clicks. Because TAFFI uses the pinch gesture to enter the tracking state, there is no obvious preferred way to emulate clicking. One approach is to map the rapid closing and opening of the thumb and forefinger to matched mouse-down and mouse-up events. This strategy has a natural, intuitive feel. Double-clicking is supported, but dragging with the mouse down is not supported.

TAFFI supports transitioning from tracking to dragging with a quick opening and closing of the thumb and forefinger (analogous to a "tap-and-a-half" gesture to drag using touchpads [9]). The dragging motion is evocative of 'readjusting' the grip on an object. This is implemented by generating a mouse-down event for the beginning of a new pinch immediately following a pinch; the corresponding mouse-up event is generated when the thumb and forefinger are separated.

Translation, Rotation and Scaling

Mouse emulation is an important feature in desktop settings, but it is perhaps more valuable to consider going beyond traditional mouse-based interactions.

Recall that the connected components algorithm can compute spatial mean and covariance of the pixels belonging to each component. This mean and covariance can be related to an oriented ellipsoidal model of the component's shape by computing the eigenvectors of the covariance matrix. The square root of the magnitude of the eigenvalues gives its spatial extent (major and minor axes size), while the orientation of the ellipse is determined as the arctangent of one of the eigenvectors, up to a 180 degree ambiguity. This ambiguity is not a problem for determining frame to frame changes in orientation.

Simultaneous changes in position, orientation and scale may be computed from the ellipsoidal model of the hole formed by the thumb and forefinger. Due to the way the thumb and forefinger of a relaxed hand typically come together, the ellipsoid is unlikely to be circular (thus having no stable orientation). Changes in scale can be used to detect movement of the hand towards and away from the camera, assuming the user holds their thumb and forefinger such that size and shape of the hole are approximately fixed, and changes in orientation are limited to the plane of the keyboard.

TAFFI uses this ellipsoidal model for one-handed navigation of aerial and satellite imagery provided by the Windows Live Virtual Earth web service. Panning of the view is accomplished by pinching and moving the hand across the keyboard, rotation by rotating the hand in the plane of the keyboard, and zooming by moving the hand up and down above the keyboard. The natural pinch-to-clutch mechanism is analogous to physically grabbing the map, and the simultaneous panning, rotation and zooming of the view gives this interaction a fluid, direct manipulation feel

Bimanual Interactions

The pinch detection technique detailed above supports the use of both hands simultaneously. Figure 2 (bottom right) illustrates the detection of two hands. In this case, a distinct hole is formed by each hand. A simple frame-to-frame correspondence strategy allows each hole to be continuously tracked as either the first or second hand.

Simultaneous tracking of multiple points of input enables a variety of bimanual interactions [5, 8]. TAFFI demonstrates bimanual input in navigating Virtual Earth imagery: the changing positions of the two tracked hands specify simultaneous changes in rotation, translation and scaling that are applied to the map view. Because the estimates are derived from the position of the hands, the two-handed technique tends to provide more stable estimates of motion than the one-handed approach. TAFFI uses the mathematical formulation presented in [12] to calculate the transforms from the movement of multiple points.

Moving both hands in the same direction pans the map view. Zooming the map view is then accomplished by pulling the two pinched hands apart, an effect analogous to stretching an elastic sheet of material. Zooming out is accomplished by moving the hands together. Moving one hand faster than the other rotates the view about an axis determined by their motion; during rotation each hand is 'pinned' to the same place on the map. As in the one-handed interface, clutching is useful in panning, rotating and zooming the view.

DISCUSSION

The pinch detection technique derives much of its value from its simplicity and robustness compared to other computer vision-based sensing strategies. However, the approach has a number of limitations.

Because it is based on connected components analysis which takes a segmented (binary) image as input, the detection of the hole formed by the thumb and forefinger depends on the quality of the segmentation. In general, seg-

mentation is very difficult, particularly in uncontrolled viewing circumstances. Mobile scenarios are particularly difficult due to rapidly changing background. Our TAFFI prototype eases segmentation by using a black keyboard, currently the style in keyboard fashion, but we note that there are many more advanced image segmentation techniques available. Further TAFFI prototypes could use active illumination such as infrared light (LEDs) to mitigate the effect of changing ambient illumination [12].

Secondly, the technique depends on the formation of a hole between the thumb and forefinger, through which the background (the keyboard) is visible. This constrains how the hand is held—no hole is formed if the user curls the other fingers underneath the thumb and forefinger. Similarly, the hand must be held at an orientation such that the hole is visible to the camera. Furthermore, the hand must be seen by the camera. The extremities of the keyboard itself may be used to indicate to the user the limits of the sensed area, but only at the height of the keyboard: the sensed volume is a *frustum*, and the sensed area thus depends on height. We have seen users go beyond the sense region when panning or zooming. It may be valuable to consider ways to design the interaction such that the user naturally stays within viewing limits, such as by providing visual or audio feedback when a pinched hand leaves the viewing frustum.

Thirdly, when used as a hand-tracker, the pinch detector may be limited by the fact that the tracked position corresponds to the center of the hole component, not the position of the point where the thumb and forefinger touch. In systems such as TAFFI where only relative motion is used, this distinction is unimportant. However, in direct manipulation frameworks, such as when the input and display are co-located, this offset may need to be addressed by further processing. It may suffice to take the position of one end of the oriented ellipsoidal model of the hole.

Finally, the technique does not support both tracking and dragging without an extra transition [4]. Much like touchpads, an additional interaction, such as our *pinch-release-pinch* transition, is necessary to support both tracking and dragging transactions with the technique. Although this is a reasonable compromise, additional usability testing is needed to evaluate the technique. Many applications do not require all three states, including perhaps many that complement the mouse and keyboard in interesting ways.

CONCLUSION

We have contributed an image processing technique to detect when the user's thumb and forefinger are touching in close-range video images. This allows user interface practitioners to realize robust interactions founded upon pinching gestures. TAFFI can simultaneously recover the position, change in orientation, and change in height of the hand above the keyboard. Unlike many other bare-hand input techniques, pinching offers unambiguous state transitions that can be used to support useful interactions beyond just tracking cursor movements. Our Virtual Earth navigation prototype demonstrates the application of pinching gestures to enable a rich set of one and two-handed gesture interactions. We would like to further explore pinch detection in scenarios where the input and display are co-located on surfaces in the real world [12], and other contexts in which it is difficult to embed touch sensing hardware.

Acknowledgements

This work grew out of discussions with Mike Sinclair, who also assisted in the construction of an early prototype. Thanks to Ken Hinckley for extensive comments.

REFERENCES

1. Balakrishnan, R. and MacKenzie, I.S., Performance differences in the fingers, wrist, and forearm in computer input control. in *Proceedings of the CHI '97 Conference on Human Factors in Computing Systems*, (1997), 303-310.
2. Baudel, T. and Beaudouin-Lafon, M. Charade: remote control of objects using free-hand gestures. *Communications of the ACM*, *36* (7). 28-35.
3. Bowman, D., Wingrave, C., Campbell, J., Ly, V. and Rhoton, C. Novel uses of pinch gloves for virtual environment interaction techniques. *Virtual Reality*, *6* (3). 122-129.
4. Buxton, W., A Three-State Model of Graphical Input. in *INTERACT '90*, (1990), 449-456.
5. Hinckley, K., Czerwinski, M. and Sinclair, M., Interaction and modeling techniques for desktop two-handed input. in *Proc. of the ACM UIST'98 Symposium on User Interface Software and Technology*, (1998), 49-58.
6. Horn, B.K.P. *Robot Vision*. MIT Press, Cambridge, MA, 1986.
7. Krueger, M.W. *Artificial Reality II*. Addison-Wesley, Menlo Park, 1991.
8. Kurtenbach, G., Fitzmaurice, G., Baudel, T. and Buxton, W., The design and evaluation of a GUI paradigm based on two-hands tablets and transparency. in *Proceedings of the CHI '97 Conference on Human Factors in Computing Systems*, (1997).
9. MacKenzie, I.S. and Oniszczak, A., A comparison of three selection techniques for touchpads. in *Proc. ACM CHI' 98 Conf. on Human Factors in Computing Systems*, (1998), 336-343.
10. Malik, S. and Laszlo, J., Visual touchpad: a two-hand gestural input device. in *Proceedings of the 6th International Conference on Multimodal Interfaces*, (State College, PA, 2004), 289-296.
11. Quek, F., Mysliwiec, T. and Zhao, M., FingerMouse: A freehand pointing interface. in *IEEE Automatic Face and Gesture Recognition*, (1995), 372-377.
12. Wilson, A., PlayAnywhere: a compact tabletop computer vision system. in *Proceedings of the 18th Annual ACM Symposium on User Interface Software Technology*, (2005), 83-92.
13. Wilson, A. and Cutrell, E., FlowMouse: A computer vision-based pointing and gesture input device. in *INTERACT '05*, (Rome, Italy, 2005).