FlyEye: Grasp-Sensitive Surfaces Using Optical Fiber

Raphael Wimmer
University of Munich
Amalienstr. 17, 80333 Munich, Germany
raphael.wimmer@ifi.lmu.de

ABSTRACT

This paper presents a method for prototyping grasp-sensitive surfaces using optical fibers. In this system one end of a fiber bundle is attached to an image sensor. The other ends of the individual fibers are attached to distinct points of a surface. Thus the image sensor can detect changes in light reception caused by a hand covering the surface. By emitting infrared light through the surface and measuring the amount of reflected light the system can also recognize touch and proximity. Mapping between pixels on the image sensor and fiber positions on the surface is generated by a relative calibration method. This setup allows to quickly build grasp-sensitive objects without electronics skills.

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation:
User Interfaces

General Terms
Human Factors

Author Keywords
grasp recognition, tangible user interface, optical fiber, computer vision

INTRODUCTION

Graspable User Interfaces, the term being coined by Fitzmaurice, Ishii and Buxton in 1995, have been a subject of scientific research for quite a while. The core of this concept are physical objects with which users interact using their bare hands. Manipulating these objects also manipulates digital information that is connected to them. It seems that most research focuses on modifying (e.g. deforming, connecting) or moving such objects. Only recently have researchers begun to explore how the way users grasp an object can be leveraged to improve human-computer interaction [2, 5, 6, 7]. Recognizing the way a user grasps an object can enhance explicit [5] and implicit [7] interaction with the object or with digital information it represents. Explicit interaction means that the user knows the meaning associated with a certain grasp type and grasps the object accordingly, like turning a mobile phone by 90 degrees to switch from call mode to camera mode. Implicit interaction means that grasps are determined by the object's touch affordances and not intended to evoke certain actions. An example for implicit interaction would be to pull out a mobile phone from a pocket, which might wake it up from standby.

For classifying and discriminating different grasp patterns several machine learning algorithms have been successfully applied so far [2, 3, 5, 6]. These algorithms do not depend on sensor technology. However, such classifiers require preprocessing of sensor data in order to deliver acceptable accuracy [5]. These methods depend on the type of sensor data, the specific application, and physical layout of sensors. The research described here focuses on sensor hardware and pre-processing of sensor data. Specifically, we present a novel way of gathering and pre-processing sensor data using computer-vision and graph-layout algorithms.

The next section shortly presents the concept behind FlyEye. The third section examines common sensor technologies for grasp-sensitive surfaces. The fourth section describes how to build grasp-sensitive surfaces using optical fiber. The fourth section describes a custom calibration method for FlyEye.

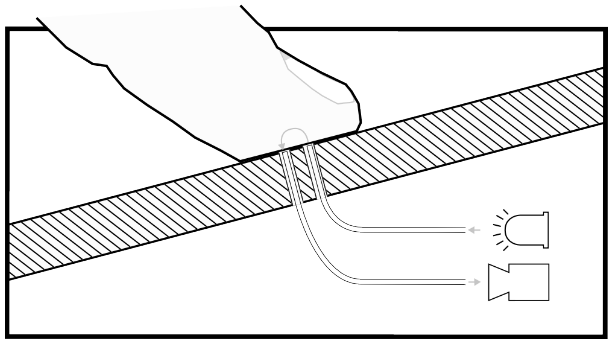

Figure 1. Basic concept of FlyEye. Infrared (IR) light is emitted from the surface via optical fibers. Objects touching the surface reflect some of the light back onto the surface. Additional fibers conduct the light to a camera. By attaching several dozen or hundred fibers to camera and IR source surfaces can be made grasp-sensitive.

FLYEYE

FlyEye is a novel method for prototyping grasp-sensitive objects. It uses optical fibers embedded into a surface to detect touches. Additional fibers emit infrared (IR) light from the surface. A camera measures the amount of IR light scattered back into the fibers (Figure 1). Computer vision and graph layout algorithms pre-process the image data for use in grasp-recognition algorithms. FlyEye has many advantages over other sensor technologies. With FlyEye the spatial resolution of the grasp-sensitive surface can be varied depending on the actual resolution needed in different areas. Unlike most other sensor technologies, FlyEye requires no electronics knowledge, thereby lowering the entry barrier. As sensing and processing do not have to happen at the point of touch, FlyEye allows very small touch-sensitive objects to be built - even smaller than with most integrated circuits. An example can be seen in Figure 2. Sensor data is processed primarily by computer vision algorithms. One can directly observe the individual processing steps and iteratively build an application-specific processing queue. FlyEye helps quickly prototyping grasp-sensitive surfaces but is also suited for permanent setups in many cases. It can replace more expensive and complex sensors in many cases. The following section presents the most common technologies for grasp-sensitive surfaces currently used and their properties.

TECHNOLOGIES FOR GRASP-SENSITIVE SURFACES

While several commercial tracking systems allow high-accuracy tracking of fingers, they require an infrastructure set up around the hand to be tracked. Such setups are not well suited for Graspable UIs as they require the user to be instrumented (magnetical, optical tracking) or are foiled by occlusion of important features (optical tracking). It is therefore equitable to augment objects with a grasp-sensitive surface. Several sensor technologies are available for making surfaces grasp-sensitive. The ones most commonly used in research prototypes are capacitive sensors [2, 5, 7]. Other prototypes use pressure sensors [3, 6], impendance sensors [4] or optical sensors [1]. A comparison of the inherent properties of these sensors (see table below) suggests that capacitive sensors are in many cases superior to other sensing techniques. This assumption is affirmed by the number of recent research prototypes using capacitive sensors compared to those using other technologies.

	cap.	imp.	opt.	press.
sensitive to human tissue	X	X		
can sense through a surface	X			
contactless	X		X	
proximity sensing	X		X	
thickness estimation	X	?		
non-ambiguous readings			X	X

While capacitive sensors are well suited for recognizing grasps, substantial knowledge of electronics and signal processing is required in order to equip a prototype with them. If capacitive sensors are to be attached to nonplanar surfaces, custom sensor arrays have to be built. For complex setups care has to be taken to shield sensors against each other and adjacent metal surfaces. Thus, prototyping grasp sensitive UIs using capacitive sensors is expensive and requires special skills. This greatly hinders iterative prototyping. SideSight[1] employs miniature infrared (IR) range-finders for detecting movement near the sides of a mobile phone. Its general mode of operation is similar to FlyEye. However, the distance information is converted to an electrical signal inside the range-finder and transmitted to a microcontroller over wires. For rapid prototyping the SideSight concept has similar shortcomings as capacitive sensors. FlyEye uses a different approach, whereby sensing and analysis of raw data is done collectively for all sensors in one place, mostly by off-the-shelf hardware. This greatly reduces implementation and maintenance overhead. The following section will describe how FlyEye works and how it can be used to quickly prototype grasp-sensitive surfaces.

FLYEYE HARDWARE

A FlyEye setup is similar to an insect's eye: One end of an optical fiber is embedded into the surface of an object, the other end is attached to a camera. Light falling onto the surface travels through the fiber and illuminates the fiber's end. The camera registers the illuminated fiber as a uniformly colored dot with a brightness proportional to the light intensity at the matching point on the surface. By spreading several dozen or hundred fibers across the surface, and bundling them at the camera end, the camera can sense light immission all over the surface.

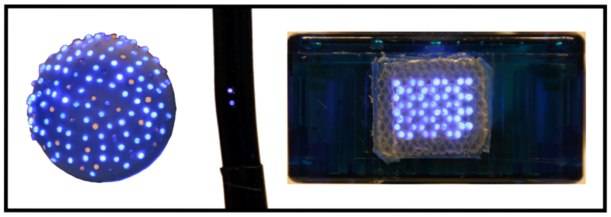

Figure 2. Some of the prototypes built so far. From left to right: grasp-sensitive ping-pong ball with 250 fibers, miniature button made of two fibers, embedded in a headphones cable, patch of 50 fibers arranged in a hexagonal grid. To enhance visibility the fibers have been illuminated.

The basic setup described so far can only determine the intensity of ambient light reaching the fibers. Therefore, this setup can not reliably discriminate between an object that is merely between light source(s) and surface and an object that is actually touching the surface. Additionally, changes in ambient light, like they may occur when rotating the object, can dramatically change the pattern of light immission at the surface. FlyEye solves these problems by means of additional fibers embedded in the surface between the other fibers. These are connected not to the camera but to an IR LED. This LED send IR light across the fibers to the object's surface. When an object approaches the surface it reflects some of this IR light back onto the surface where it travels through the adjacent fibers to the camera, equipped with an IR filter (see Figure 1). The closer the object gets the more light is reflected back onto the surface. This works well for finger tips as the IR light enters the skin and illuminates it from the inside. Thus, even when the finger is in direct contact with the surface IR light from the fibers is scattered back into the sensing fibers. By using IR light the influence of visible light sources is almost completely eliminated. However, IR emitters - the sun being a major one - still light fibers that are not covered by a finger tip, resulting in erroneous detection of touches. To mitigate this effect

FlyEye uses modulated IR light combined with background subtraction. Only every second frame the camera captures is illuminated by the IR LED. The other frames are captured without active IR emission. By subtracting an unlit frame from the following lit frame a difference image is obtained that only shows light reflected back from objects (Figure 4, upper left). The smaller the distance between object and surface gets the brighter the corresponding fibers appear in the difference image. These differences in brightness can be used to gather low-resolution proximity information.

For evaluating the concept, several prototypes were built. Figure 2 shows some of them. Most prototypes use 1mm optical fiber. The miniature button shown in the middle uses 0.5mm fiber. A Point Grey Firefly MV black and white video camera with an IR filter captures images from the end of the fiber bundle. The Firefly can be programmed to emit a strobe signal on every frame it captures. Unlike higher-end cameras it does not support a strobe signal only on every second frame, however. Therefore a 4518 counter IC is used to trigger a BC107B transistor on every second strobe signal. The transistor controls a SFH485 IR LED which is attached to the end of the illuminating fiber bundle. For quicker prototyping one end of each individual fiber was flattened with a hot soldering iron so it resembles a nail's head. This allows for either glueing the fiber's end to the inside of a transparent surface or to push the fiber through holes drilled into the surface. In the latter case the nail's head prevents the fiber from slipping through the hole (Figure 3). If the surface needs to be smooth the head can be cut off after glueing the fiber to its hole. However, the head increases the light-receiving surface of the fiber, thereby increasing the sensitivity. Therefore, the heads were not removed in most prototypes. A thin layer of hot glue was used to fill the gaps between the heads in this case. In several cases some fibers were defective, not conducting enough light. Due to the large number of remaining fibers these were discarded from the image analysis.

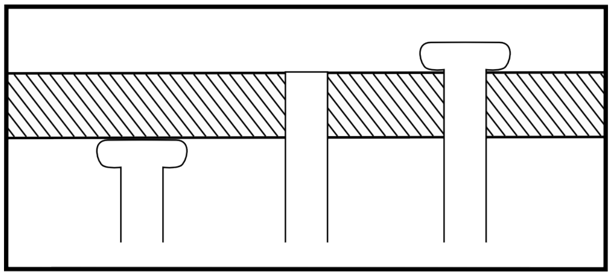

Figure 3. Fibers can be attached to the surface in different ways. From left to right: glueing fibers with nail heads to the inner side of a transparent surface, glueing fibers into holes drilled into the surface, mounting fibers with nail heads in holes drilled into the surface.

CALIBRATION

The flexible arrangement of fibers on the surface requires additional pre-processing before the sensor data can be fed to a grasp classifier. In the following we show how standard algorithms from the domains of computer-vision and information visualization can be used to extract meaningful data from the camera frames.

Challenge

Owing to the unconstrained fiber placement the bundled fiber ends are rarely in the same order as the fiber ends that are attached to the surface. This means that two fibers that are adjacent on the surface may not be adjacent in the bundled end. This poses a problem when trying to apply machine-learning algorithms to the image data: Low-dimensional features (combinations of values) are better suited for machine learning classifiers than high-dimensional ones [5]. For FlyEye this means to transform grasp information described as ”which of the pixels are lit” to ”where on the surface are lit areas”. The latter representation usually has a lower dimension than the raw image data as multiple lit fibers can be aggregated into one lit area. This allows for better feature reduction as this information can be stored either as a variable-resolution grid of lit areas or as a list of X-Y coordinates of lit areas. The grid's resolution can be tuned to balance recognition accuracy and robustness.

Absolute Mapping

In order to convert the image data to such a list or grid representation, for every fiber its relative surface position with regard to the other fibers has to be determined. This could be done by noting the position within a reference coordinate system for every fiber and determining which surface position belongs to which position in the camera image. Instead of measuring each fiber by hand moving light patterns could be projected onto the surface. This requires a fixed, calibrated setup of object and projector. In order to reach the whole surface of a convex object, the objects's placement has to be adjusted after each pass. Additionally, certain cavities in the surface may be hard or impossible to reach with projected light. The required setup is hardly suitable for rapid prototyping. Instead of this absolute mapping method FlyEye uses a fast, relative method.

Relative Mapping

The relative mapping algorithm for FlyEye utilizes the fact that for grasp recognition not the absolute surface position of each fiber has to be known but only its position relative to its neighbors. The algorithm can be divided into three steps, namely Fiber Detection, Finding Neighbors, and Untangling (Figure 4). These steps can be partially executed in parallel.

Fiber Detection For the algorithm to detect the position of the fibers in the camera image, every fiber (or multiple simultaneously) needs to be touched once. This can be done by sliding a finger or other object across the surface. The difference image is treated with a thresholding filter and a smoothing filter. All fibers that are covered by a finger appear as white circles. A circular Hough transform is used to detect such circles, returning a list of circle centers and radii.

Finding Neighbors Once a fiber's position in the camera image is known, the algorithm searches for fibers that are adjacent to it on the surface. Provided that only a single finger or other small object touches the surface in the calibration phase, it can be assumed that all fibers that are lit in a single camera image are adjacent to each other. This adjacency information is stored in an undirected graph. By

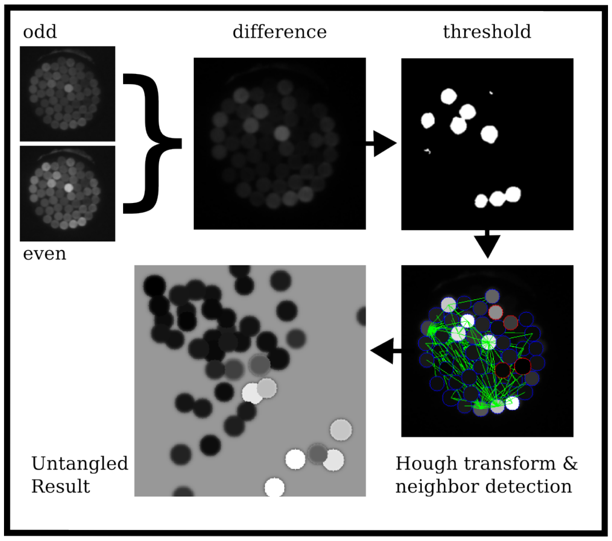

Figure 4. Complete processing chain. The unlit frame is subtracted from the frame that is lit by the IR light source. A threshold filter extracts fibers that are touched. A circular Hough transform extracts the positions of all fiber ends in the image. After the neighbors of each fiber are found a force-directed algorithm tries to reconstruct the fiber arrangement at the surface. Bright fibers are now clustered together.

moving the finger across the whole surface area a graph linking all fibers is built. The size of the finger tip with regard to the spacing between fibers on the surface determines the quality of the resulting graph. If the finger tip is smaller than the surface distance between two adjacent fibers there will always be only one fiber lit. Thus, no graph can be generated. If the finger tip is much wider than the fiber spacing on the surface then fibers that are not directly adjacent will be lit at the same time. Thus the graph contains incorrect edges. A good compromise is a finger tip width that is between once and twice the distance between adjacent fibers. As obtaining fingers of a certain width is difficult a plastic rod with appropriate width was used instead.

Untangling The nodes of the graph obtained in the previous step are then layed out in two dimensions. This is done by a Fruchterman-Reingold force-directed algorithm (also known as spring layout) that tries to arrange the nodes so that their edges overlap as little as possible. The algorithm creates a node arrangement that mirrors the arrangement of the fibers on the surface. The distance between the nodes is not necessarily proportional to the surface distance between the associated fibers, however. The node positions generated by this algorithm allow aggregating fibers with the same brightness or interpolating between fibers of different brightness. It should be noted that the Fruchterman-Reingold-algorithm is non-deterministic. Thus, the final node positions will vary between different runs of the algorithm, requiring re-training of the classifier.

For the FlyEye prototype this algorithm was implemented in Python using OpenCV[1] for image acquisition and processing and NetworkX[2] for the Fruchterman-Reingold-Layout.

[1]http://opencv.willowgarage.com/wiki/
[2]http://networkx.lanl.gov/

CONCLUSION AND FUTURE WORK

We have presented a flexible method for prototyping grasp-sensitive surfaces. This method does not require electronics skills or expensive equipment. It is well suited for small- to medium-scale prototypes or individual setups. Besides this application the proposed methods are very well suited for other domains. Applying computer-vision and graph-layout algorithms to this problem proved to be very effective in pre-processing the sensor data. The proposed relative calibration method is not restricted to FlyEye but can be applied to other sensor data where the physical location of the sensor is not fixed or unsure. The fibers can also be used to augment a surface with optical feedback by coupling visible light into the fiber bundle. Once the positions of the fibers are determined by the relative calibration method a projector could actually display images on the surface by projecting a pre-warped image onto the bundled end of the fibers.

Source code available at:
http://www.medien.ifi.lmu.de/team/raphael.wimmer/projects/FlyEye/

REFERENCES

1. A. Butler, S. Izadi, and S. Hodges. Sidesight: multi-"touch" interaction around small devices. In *Proceedings of UIST '08*, pages 201–204, New York, NY, USA, 2008. ACM.
2. K. Kim, W. Chang, S. Cho, J. Shim, H. Lee, J. Park, Y. Lee, and S. Kim. Hand Grip Pattern Recognition for Mobile User Interfaces. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 1789. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.
3. P. G. Kry and D. K. Pai. Grasp recognition and manipulation with the tango. In *International Symposium on Experimental Robotics*, volume 10. Springer, 2006.
4. J. Mäntyjärvi, K. Nybergh, J. Himberg, and K. Hjelt. Touch Detection System for Mobile Terminals. In *Proceedings of MobileHCI '05*. Springer, 2004.
5. B. T. Taylor and M. V. Bove. Graspables: grasp-recognition as a user interface. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 917–926, New York, NY, USA, 2009. ACM.
6. R. N. J. Veldhuis, A. M. Bazen, J. A. Kauffman, and P. H. Hartel. Biometric verification based on grip-pattern recognition. In E. J. Delp and P. W. Wong, editors, *Security, Steganography, and Watermarking of Multimedia Contents*, volume 5306 of *Proceedings of SPIE*, pages 634–641. SPIE, 2004.
7. R. Wimmer and S. Boring. HandSense - Discriminating Different Ways of Grasping and Holding a Tangible User Interface. In *Proceedings of the Third International Conference on Tangible and Embedded Interaction (TEI'09)*, February 2009.