SpeckleEye: Gestural Interaction for Embedded Electronics in Ubiquitous Computing

Alex Olwal
MIT Media Lab, 75 Amherst St, Cambridge, MA
olwal@media.mit.edu

Andy Bardagjy
MIT Media Lab, 75 Amherst St, Cambridge, MA
bardagjy@media.mit.edu

Jan Zizka
Comenius University DAI, Mlynska dolina, Bratislava, Slovakia
zizka@media.mit.edu

Ramesh Raskar
MIT Media Lab, 75 Amherst St, Cambridge, MA
raskar@media.mit.edu

http://specklesense.media.mit.edu

CHI'12, May 5–10, 2012, Austin, Texas, USA.
ACM 978-1-4503-1016-1/12/05.

Abstract
We introduce SpeckleEye, design and implementation of an embedded gesture and real-time motion tracking system using laser speckle. SpeckleEye is a low-cost, scalable, open source toolkit for embedded speckle sensing and gestural interaction with ubiquitous devices in the environment. We describe embedded speckle sensing hardware and firmware, a cross-platform gesture recognition library optimized to run on embedded processors, and a set of prototypes that illustrate the flexibility of our platform.

Author Keywords
Gestures, laser speckle, embedded, input devices.

ACM Classification Keywords
H.5.2 [Information Interfaces And Presentation]: User Interfaces - Graphical user interfaces; Input devices and strategies; Interaction styles;

General Terms
Design, Human Factors, Experimentation.

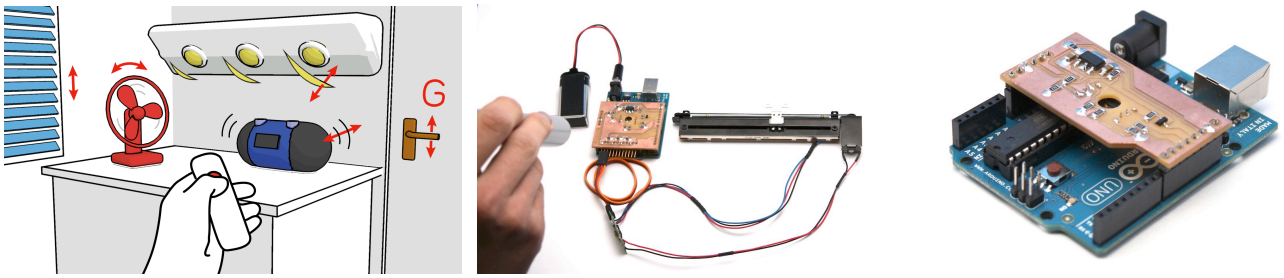

Figure 1. (Left) SpeckleEye brings low-cost motion sensing to embedded electronics and mechanical devices. This could allow gesture-based control of devices in a home, such as window blinds, fans, stereos, lighting or doors. (Center) SpeckleEye senses projected laser speckle from a handheld speckle projector, to actuate devices. (Right) Speckle is sensed using embedded low-resolution, low-power sensors that are significantly faster and smaller than conventional cameras. SpeckleEye can be directly integrated and accessed from widely available microcontrollers using our software libraries.

Introduction

As computation, sensing, and display are becoming ubiquitous in our physical environments, the exploration of new forms of interaction become increasingly relevant. While we could interact with this distributed technology using integrated sensors or remote controls, many scenarios benefit from explicit gestural interaction. The importance of pointing for spatial referencing has, for example, inspired research in gestural interaction devices.

While on-board sensors may efficiently sense motion, they typically rely on paired communication. The XWand [8], for example, uses RF communication to transmit user input and motion derived from on-board accelerometers, magnetometers, gyroscopes and physical buttons, while an IR LED enables position tracking from external cameras. Cameras can also be used to track laser pointer cursors in interactive applications [2, 5]. While camera-based systems are attractive for motion tracking, their cost, computational demands, latency, accuracy and power requirements limit scalability and application in embedded systems.

Approaches, such as RFIG [4] and LightSense [3], utilize embedded photosensors activated by light from user-controlled devices, to create environments with multiple local, low-resolution, embedded sensors with dedicated special-purpose electronics. Photosensors are also often embedded in devices to sense ambient light and receive time-multiplexed commands from remote controls. They are suitable for transmitting 1D serial data, but less appropriate for motion input.

High-frequency speckle patterns, generated from the interference of coherent light [1], on the other hand, can be used to track fast motion using minimal hardware. We have recently described an exploration of

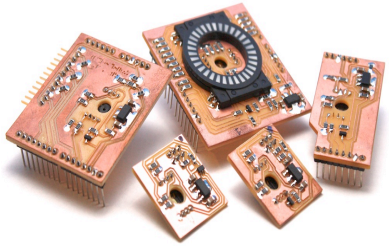

Figure 2. Different generations, form factors, and interfaces for our microcontroller-compatible laser speckle sensing modules. (Left) Exposed PWM ports for servo control. (Center) Integrated ring LED light. (Right) Minimal pin coverage to fit onto an Arduino's SPI ports. (Bottom) Minimal design with female ICSP header.

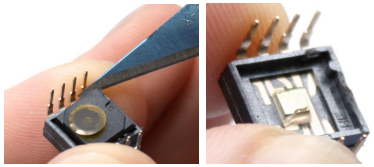

Figure 3. For speckle sensing, the field-of-view of the sensor is increased by removing a limiting pinhole aperture.

laser speckle sensing and its applications for user interaction using high-speed cameras and modified lensless optical mice connected to computers [7].

This paper extends our previous work to microcontroller-compatible hardware that provides embedded optical flow calculations and image analysis to track motion significantly faster and more precisely than classical computer vision systems.

We describe the design and implementation of a practical platform to enable embedded gestural interaction on resource-limited hardware using low-cost sensing of laser speckle projected from handheld pointing devices (Figure 1). We introduce a speckle sensing platform with hardware that is plug-and-play compatible with widely available, low-power microcontrollers. We also provide accompanying open source firmware and a gesture recognizer implementation that has been optimized to run on resource-limited microcontrollers.

Embedded Speckle Sensing

Our approach is based on the use of handheld pointing devices, *Speckle Projectors*, which project laser speckle into the environment. The speckle motion is sensed with small low-power sensors that can be integrated to control existing technologies. Three-dimensional relative motion is then inferred from speckle displacement, and consecutive translations can be analyzed to enable gestural interaction.

Handheld Speckle Projectors. Speckle projectors are constructed by inserting a diffuser in front of a laser. The laser intensity and diffuser angle affect its range and field-of-view. In these experiments, we use a 50mW 780nm laser diode with an 80° diffuser, providing approximately a three-meter range in indoor ambient lighting conditions.

While engineered holographic diffusers can be used to specify scattering amount and angle, diffusers can also be constructed using vellum, tracing paper or diffuse tape.

Integrated Sensor. Optical sensors in modern mice are particularly suited to detecting image translations, such as the rapid motion of our speckle pattern. Furthermore, optical mouse sensors are inexpensive, designed for embedded system integration, suitable for battery-powered applications, and significantly faster and smaller than conventional cameras.

For speckle sensing, we selected the Avago ADNS-3000. It is widely available, combines high performance with ultra-low power consumption, and uses the standardized Serial Peripheral Interface protocol. The sensor is available in an 8-pin through-hole IC and contains a 22×22 CMOS photodetector array. The chip computes optical flow in two axes, minimum, maximum and average pixel values, exposure time, and surface feature count.

Microcontroller Interface. Several circuit board designs and interfaces were developed to enable plug-and-play integration with common microcontrollers. These designs include a sensor, voltage regulator and level translation circuitry. Some have been customized for particular applications, with motor, servo, or lighting control peripherals. To further simplify prototyping of speckle sensing applications, we developed Arduino-compatible (http://www.arduino.cc/) shields that carry

the sensor and interface electronics (Figures 1 and 2). A companion library to communicate with the ADNS-3000 was also developed. Both the shield and library are released under open source licenses [6]. The circuit boards were designed using the freeware version of EagleCAD, and fabricated by milling copper-clad FR1 boards on a Roland MODELA MDX-15.

Motion Sensing and Distance Estimation

The sensor continuously analyzes acquired images and calculates optical flow, magnitude, and direction of 2D translations using dedicated processing hardware. The sensor can be configured to report this data as 8- or 12-bit integers that map to lateral motion along the vertical and horizontal axes.

Laser speckle size and density are proportional to the distance of the projector from the sensor. Interfacing directly with the optical sensor exposes internal data, which can be analyzed to approximate the projector's distance to the sensor.

Shutter opening time. The sensor's automatic gain adjusts the shutter based on overall brightness, which increases when the speckle projector is moved closer to the sensor. The value is reported as a 16-bit integer that represents the opening time in clock cycles.

Average pixel intensity. The sensor also computes the average value of all pixels in the image. The average pixel intensity increases with the speckle projector's proximity to the sensor.

Trackable Features. The number of valid features that the sensor can use for tracking is reported as a measure of surface quality. Since the speckle is projected, larger distances yield larger speckle, and result in fewer trackable features.

Gesture Recognition

To support continuous freeform gestures, an optimized version of the $1 gesture recognizer [9] was implemented in cross-platform C++. The 16 templates from the original paper are used, but resampled as 64 uniformly distanced integer points, normalized in the range [-4095, 4096]. The algorithm processes a continuous stream of points and upon a trigger (e.g., a timeout for no received data) the path is compared to a set of stored templates. This computation produces a list of distances, where the template with the smallest distance to the observed path is chosen as the recognized gesture.

Our implementation of the algorithm allows this library to run on resource-limited microcontrollers. In our rewritten library we avoid dynamic memory allocations and reuse all large memory blocks in subsequent computations, making memory use predictable for platforms with limited resources. C++ templates enable the use of different data types (e.g., integers or floating point) for the gesture data format, which allows tradeoff between memory footprint and precision. We use integer representation on our microcontrollers (2 bytes), which requires less storage and calculations compared to floats (4 bytes on our platform).

Prototypes and Applications

To explore the potential of our toolkit, several prototypes were developed. In our prototypes, sensed motion data is processed and can be directly mapped to output devices or, for more sophisticated interaction, our gesture recognition library can be employed.

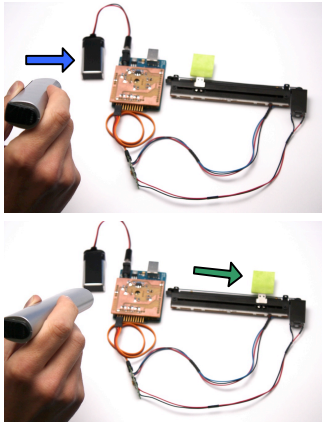

Figure 4. The linear actuator is mapped to the pointing device's horizontal movement, which provides responsive mapping and direct control.

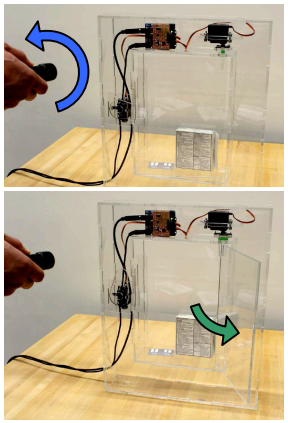

Figure 5. The cabined is opened with a counter-clockwise gesture to unlatch and open the door, while a clockwise gesture reverses the action.

Complex embedded interactions can thus be quickly prototyped using our libraries and a few lines of code.

Direct Control of Mechanical Devices. The user's motion can be mapped to different electromechanical actuators, such as linear actuators or servo motors, for direct and responsive control. (See Figure 4.) These mappings are, for example, applicable to situations where the user wishes to control devices in hard-to-reach or hazardous areas.

Gestural Unlocking. The door to our cabinet prototype can be operated using "secret" user-defined gestures that trigger different motion sequences for the two embedded servos that control the latch and door rotation. In our example implementation, a counter-clockwise gesture unlatches the door and swings it open, while a clockwise gesture reverses the action. (See Figure 5.)

Interfacing with Commercial Products. Our toolkit makes it possible to operate existing consumer electronics by controlling the supply voltage, for example, using servo boards, transistors or relays. Figure 6 shows how gestural input to a microcontroller determines velocity and direction for a small fan. This type of interface may be applicable to ceiling fans or other devices that are out of reach, but would benefit from direct pointing control. Figure 7 shows a servo-mounted camera, which will rotate clockwise or counterclockwise depending on the input gesture. Furthermore, the camera is running the Canon Hack Development Kit (http://chdk.wikia.com/wiki/CHDK), which allows us to trigger the shutter with a custom script and a modified USB cable attached to a digital output pin on the microcontroller. Thus, when the user is satisfied with the camera's orientation, a "V" gesture can be used to stop the rotation and take a picture.

Future Work

Multi-user. Multiplexing speckle projectors in time, spectra, or modulation frequency domains could enable multi-user interaction, which may be beneficial to robot localization, motion capture, collaborative interaction, and music creation, for example.

Communication and Feedback. Not all forms of interaction lend themselves to gestural interaction (e.g., text input). In such a case, a hybrid device incorporating multiple input modalities could be ideal. Modulating the output of the speckle projector could encode this information. Alternatively, a speckle projector could be embedded within a conventional remote control. With bidirectional communication, additional feedback, such as vibrotactile could be incorporated.

Absolute Positioning and Rotation. The speckle sensor detects relative translation in the lateral plane, and with some secondary statistics, depth can be approximated. Estimation algorithms combined with additional speckle sensors and projectors may allow inference of absolute rotation and position.

Feedback. For mid-air interaction, feedback is critical. Informal experiments indicate that interaction is improved by attaching a visible laser coaxially with the speckle projector. It provides users both with visual feedback of the gesture path and helps them constrain motion to the lateral plane relative to the sensor. Currently, we provide feedback using LEDs when speckle motion is sensed. In the future, other

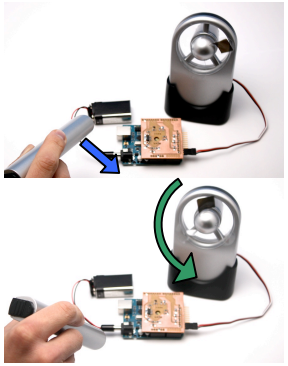

Figure 6. The microcontroller drives a small fan via a servo board. Here, the user has continuous velocity control through vertical movement.

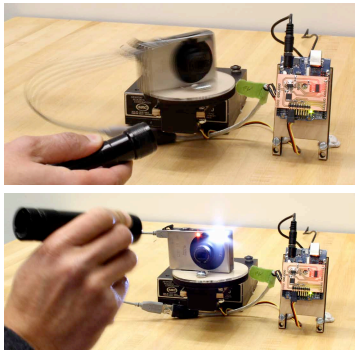

Figure 7. Circular gestures from a distance rotate (top) a servo-mounted camera, while a "V" gesture takes a photo (bottom).

modalities such as audio or vibrotactile could be considered.

Conclusions

As computing and embedded devices progressively permeate our environment, interacting with those artifacts becomes increasingly important. There are many scenarios where interacting using gestures may be the most appropriate interaction method. Camera and computer-based systems are traditionally used for optical motion sensing, but issues such as cost, speed, size and power consumption have largely prevented widespread adoption of gestural interaction for embedded devices.

In this work, we present SpeckleEye, a scalable solution for embedded speckle sensing and gestural interaction with ubiquitous devices in our environment. Our toolkit includes open source hardware designs and software libraries, including a cross-platform gesture recognition library optimized to run on microcontrollers.

To explore the potential of our toolkit, we have constructed several prototypes to illustrate the flexibility of our platform. These prototypes demonstrate gestural interaction for rotary motion, linear translation, and control of off-the-shelf devices.

With this work we hope to inspire interaction designers, researchers, and engineers to further explore expressive interaction with this novel sensing technique in everyday devices.

Acknowledgements

Alex Olwal was supported by a Swedish Research Council Fellowship and a Blanceflor Foundation Scholarship, Andy Bardagjy was supported by the NSF Graduate Research Fellowship, and Jan Zizka was supported by Comenius University and ME-Inspection.

References

1. Goodman, J. W. 2007. Speckle Phenomena in Optics: Theory and Applications. Roberts and Company, Englewood.
2. Olsen, Jr, D.R. and Nielsen, T. 2001. Laser Pointer Interaction. Proc. *CHI '01*. ACM, New York, NY, USA, 17-22.
3. Olwal, A. 2006. LightSense: Enabling Spatially Aware Handheld Interaction Devices. Proc. *ISMAR '06*, 119-122.
4. Raskar, R., Beardsley, P., van Baar, J., Wang, Y., Dietz, P., Lee, J., Leigh, D., and Willwacher, T. 2004. RFIG Lamps: Interacting with a Self-describing World via Photosensing Wireless Tags and Projectors. Proc. *SIGGRAPH '04.*
5. Rekimoto, J. and Saitoh, M. 1999. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. Proc. *CHI '99*. 378-385.
6. SpeckleSense. http://specklesense.media.mit.edu/. February 2012.
7. Zizka, J., Olwal, A., and Raskar, R. SpeckleSense: Fast, Precise, Low-cost and Compact Motion Sensing using Laser Speckle. Proc. *UIST '11*, 489-498.
8. Wilson, A. and Shafer, S. 2003. XWand: UI for Intelligent Spaces. Proc. *CHI '03*, 545-552.
9. Wobbrock, J. O., Wilson, A.D., and Li, Y. 2007. Gestures without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes. Proc. *UIST '07*, 159-168.