Multi-camera Microenvironment to Capture Multi-view Time-Lapse Videos for 3D Analysis of Aging Objects

Lintao Guo, Hunter Quant, Nikolas Lamb, Benjamin Lowit, Natasha Kholgade Banerjee, and Sean Banerjee(✉)

Clarkson University, Potsdam, NY 13699, USA
{linguo,quanthd,lambne,lowitbp,nbanerje,sbanerje}@clarkson.edu

Abstract. We present a microenvironment of multiple cameras to capture multi-viewpoint time-lapse videos of objects showing spatiotemporal phenomena such as aging. Our microenvironment consists of four synchronized Raspberry Pi v2 cameras triggered by four corresponding Raspberry Pi v3 computers that are controlled by a central computer. We provide a graphical user interface for users to trigger captures and visualize multiple viewpoint videos. We show multiple viewpoint captures for objects such as fruit that depict shape changes due to water volume loss and appearance changes due to enzymatic browning.

Keywords: Multi-camera · Time-lapse · Multiple viewpoint Time-varying

1 Introduction

Natural and man-made objects exhibit time-varying transformations at varying time scales. An ice cube melts in a few minutes when left outside, while a banana darkens over several days, and an iron pipe corrodes over several months. While there exist approaches to study time-varying changes to appearance in objects [3,5,7,8], approaches to model 3D time-varying shape are limited to human faces and bodies [1,2] or use turntables with single scanner setups that may introduce unwanted deformations to the object [6]. As discussed in [4], we provide an approach to reconstruct spatiotemporal 3D models of time-varying shape and appearance changes in fruit using multi-view time-lapse videos. In this paper, we describe our work on providing multi-camera microenvironments to capture multi-view time-lapse videos.

The microenvironment images objects from multiple viewpoints using a set of Raspberry Pi v2 cameras controlled by corresponding Raspberry Pi v3

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-319-73600-6_37) contains supplementary material, which is available to authorized users.

K. Schoeffmann et al. (Eds.): MMM 2018, Part II, LNCS 10705, pp. 381–385, 2018.
https://doi.org/10.1007/978-3-319-73600-6_37

computers. Our microenvironment uses modularized hardware to install the Raspberry Pi cameras and computers that can be printed on a consumer grade 3D printer. We provide a Java graphical user interface to capture and visualize multi-viewpoint time-lapse captures. Using the microenvironment, users can perform time-lapse captures of tabletop objects showing shape and appearance changes over time with customized lapse durations and capture lengths.

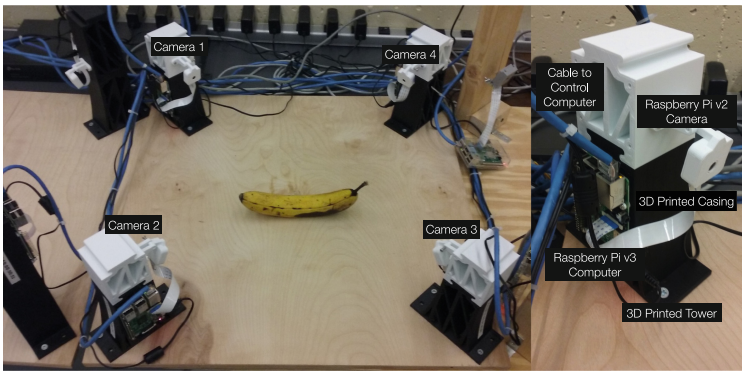

Fig. 1. Microenvironment to capture multi-view time-lapse images consisting of four Raspberry Pi v2 cameras connected to four Raspberry Pi v3 computers.

2 Multi-camera Microenvironment

As shown in Fig. 1, our microenvironment consists of a 24" × 24" plywood base with four 3D printed adjustable height camera towers. We modularize the camera towers to allow the user to to raise or lower the height of the cameras by adding or removing tower pieces. Each camera tower consists of a Raspberry Pi v3 computer and a Raspberry Pi v2 camera. The Raspberry Pi computers are connected to an Asus ESC4000-G3 central computer using a Gigabit Netgear switch. We use the central computer to start and stop captures, and to store and visualize the captured data.

The user interacts with the microenvironment using a graphical user interface (GUI) on the central computer implemented using the Swing package in Java. Our interface provides the user with a "Capture" and a "View" tab, as shown in Fig. 2. To capture image sequences, the user navigates to the "Capture" tab. As shown in Fig. 2(a), we allow the user to begin capturing image sequences after all cameras have been tested and synchronized. During testing, we capture a sample image to ensure the cameras are functional. We mark non-functional cameras in red to enable rapid identification and replacement. During synchronization, we update the local time on each Raspberry Pi computer to the time on the central computer to reduce time drift.

As shown in Fig. 2(b), the user specifies the number of images and capture interval and presses "Capture" button on our interface to start a capture. We send a parallelized capture signal to the four Raspberry Pi computers at the user specified capture interval. We store the image captured by each Raspberry Pi camera on its corresponding Raspberry Pi computer. We transfer all images to the central computer before triggering the capture of images at the next time-lapse instant. The capture stops once all images have been captured or if the user interrupts the capture by click the "Stop" button.

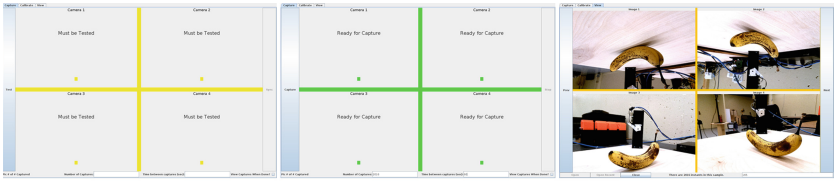

Fig. 2. Java based graphical user interface to allow users to (a) test and synchronize cameras, (b) set up a capture, and (c) visualize capture results.

To visualize a capture the user uses the "View" tab on the interface to load time-lapse image sequences from the multiple viewpoints as shown in Fig. 2(c). Once the capture is loaded, we provide the user with the number of images in the capture sequence, and "Prev" (or 'previous') and "Next" buttons to navigate through the image sequence. The user navigates to a picture by entering the image index and pressing enter, by scrolling forward and backward using the "Prev" and "Next" buttons, or by using the left and right arrow keys on the keyboard. We provide keyboard shortcuts "X" and "D" to the user to export and delete images from the capture sequence. Figures 3, 4, and 5 show multi-view time-lapse captures for a banana, a green bell pepper, and a Gala apple.

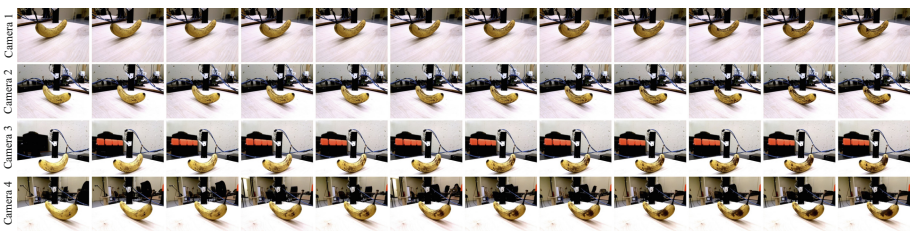

Fig. 3. Banana blackening over one week at 5 min intervals.

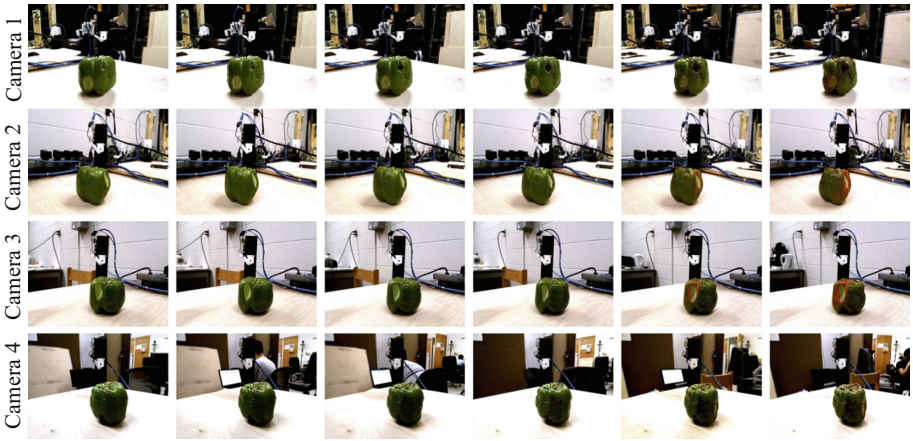

Fig. 4. Green bell pepper shrinking and reddening over five days at 5 min intervals. (Color figure online)

Fig. 5. Apple browning over five days at 2 min intervals.

3 Future Work

In future work, we will integrate calibration of the geometric and photometric properties of the cameras using a variety of calibration targets into the GUI. We will also include controls for temporal operations found in traditional multimedia viewers. To accommodate increasing camera counts in the microenvironment, we will use network latency and shutter lag statistics to minimize differences in capture times at each frame for captures of phenomena occurring on the order of seconds to a few minutes such as melting candles, cooking food, insect motions, and movements of mechanical parts. Future work will include providing fine-grained control over individual camera parameters such as exposure compensation, white balance gains, region of interest, and ISO, and providing the option to switch between JPEG files and the original RAW images.

Acknowledgements. This work was partially supported by the National Science Foundation (NSF) grant #1730183.

References

1. de Aguiar, E., Stoll, C., Theobalt, C., Ahmed, N., Seidel, H.P., Thrun, S.: Performance capture from sparse multi-view video. ACM Trans. Graph **27**(3), 98 (2008)
2. Beeler, T., Hahn, F., Bradley, D., Bickel, B., Beardsley, P., Gotsman, C., Sumner, R.W., Gross, M.: High-quality passive facial performance capture using anchor frames. In: SIGGRAPH (2011)
3. Enrique, S., Koudelka, M., Belhumeur, P., Dorsey, J., Nayar, S., Ramamoorthi, R.: Time-varying textures: definition, acquisition, and synthesis. In: SIGGRAPH Sketches (2005)
4. Guo, L., Quant, H., Lamb, N., Lowit, B., Banerjee, S., Banerjee, N.K.: Spatiotemporal 3D models of aging fruit from multi-view time-lapse videos. In: MMM (2018)
5. Langenbucher, T., Merzbach, S., Möller, D., Ochmann, S., Vock, R., Warnecke, W., Zschippig, M.: Time-varying BTFs. In: CESCG (2010)
6. Li, Y., Fan, X., Mitra, N.J., Chamovitz, D., Cohen-Or, D., Chen, B.: Analyzing growing plants from 4D point cloud data. ACM Trans. Graph **32**(6), 157 (2013)
7. Lu, J., Georghiades, A.S., Glaser, A., Wu, H., Wei, L.Y., Guo, B., Dorsey, J., Rushmeier, H.: Context-aware textures. ACM Trans. Graph **26**(1), 3 (2007)
8. Sun, B., Sunkavalli, K., Ramamoorthi, R., Belhumeur, P.N., Nayar, S.K.: Time-varying BRDFs. TVCG (3), 595–609 (2007)