CS348K: Visual Computing Systems
Class/Reading Response Template

Reminder: please make sure the PDF you submit to Gradescope DOES NOT have your name on it. We will concatenate all responses and give everyone in the class a PDF of all responses.

Part 1: Top N takeaways from discussions in the last class. Note: this part of the response is unrelated to the current reading, but should pertain to the discussion of the prior reading in class (or just discussion in the class in general, if there was no reading):

- What was the most surprising/interesting thing you learned?
    - I think the idea of HDR is really cool and breakthrough. It aligns multiple images of the same scene, each taken with the same exposure setting.
- Is there anything you feel passionate about (agreed with, disagreed with?) that you want to react to?
    - The alignment. Simply adding a bunch of frames will likely to get a blurry image. We should ffirst align and then average these frames.
- Did class cause you to do any additional reading on your own? If so, what did you learn?
    - In this paper, the aligning is done by comparing pixel values. Since similar exposures have similar sharpness, similar noise characteristics, it would be easier to do the alignment. However, for frames with different exposure settings, the main challenge is that they may have different brightness and contrast levels, as well as differences in motion blur or other temporal artifacts. One solution is to use the Lucas-Kanade (LK) algorithm. It is for optical flow estimation, which refers to the apparent motion of objects in an image. The LK algorithm first detects a set of feature points, then tracks the detected feature points in the subsequent frames. It assumes that the motion of each point can be approximated by a small translation, the brightness remains constant between the frames, and then estimates the small translation by solving linear equations. Finally it solves the equations to get the small translation statistically using least-squares method. By estimating the optical flow between pairs of frames, we can estimate the motion of objects in the scene and use this information to align the frames.
- Major takeaways in general?
    - Four design principles. Immediate, automatic, natural, conservative.
    - Take a bunch of photographs and adding them together will not benefit anything,

Part 2: Answers/reactions to instructor's specific prompts for this reading. (Please see course website for prompts).

I'd like you to describe the major pieces of the Frankcamera abstract machine (the system's nouns): e.g., devices, sensors, processors, etc. Give some thought to why a sensor is not just any other device? Is there anything special about the sensor?

Devices: Devices includes cameras, lens, a flash, and other assorted devices. Devices are controlled by the sensor pipeline, and their actions can be coordinated for precise timing.

Sensors: One important characteristic is stateless. A sensor pipeline transforms requests into frames by configuring the hardware. The frames produced by the sensor pipeline are queued and retrieved asynchronously by the application. The sensor pipeline is the manager of the while image pipeline.

Processors: The image processer sits between the raw output of sensorw and the application processor. It 1. generates useful statistics from the raw image data, and 2. transforms image data into requested format.

The special things are 1. the sensor is not just any other device because it is the manager of the pipeline, responsible for configuring the hardware necessary to produce the frame. Also, 2. accurate labeling of the frame produced by the sensor is essential for image processing.

That's also why a sensor is not just any other device. Unlike lens, which are peripheral to the image processing process, the sensor is the raw source of image data.

Then describe the major operations the machine could perform (the system's verbs). For example, in your own words, what is a "shot"? Would you say a shot is a command to the abstract machine? Or is a shot a set of commands? What do you think about the word "timeline" as a good word to describe what a shot is "shot"?

Shot: Shot refers to a command sent to the camera system to capture a single image. It is a command, not a set of commands. I think the term "timeline" could be a good word to describe a shot because it implies a sequence of events that occur over a period of time, and a shot in the Frankencamera system is indeed a sequence of the image capture.

Sensor: A sensor is not a command to the abstract machine, but it contains two operations, capture and stream. The two major operations of a sensor are to capture a single frame or to stream a continuous feed of frames.

Frame: The getFrame operation can retrieve a single frame from the camera sensor. This operation sends a command to the camera to capture a new frame and returns the resulting image as a Frame object.

Devices: Lens have setFocus, setZoom, and setAperture operations. The setFocus operation allows the user to specify a specific focus distance for the len. The setZoom operation sets focal length for the lens to zoom to. The setAperture operation allows the user to specify a specific aperture value, which controls the amount of light that passes through the lens and reaches the sensor or film.
For the flash, it has a single operation fire that tells it to fire with a specified brightness and duration.

What output does executing a shot generate? How is a "frame" different from a shot? Why is this distinction made by the system?

Executing a shot should generate oneframe.

Shot is asynchronized while frame is synchronized, so that shot allows greater parallelism in image capture and post-processing operations.

This distinction is because that the getFrame method is the blocking call, so that it can't perform other tasks while waiting for the function to complete

Would you say that F-cam is a "programmable" camera architecture or a "configurable architecture". What kinds of "programs" does the abstract machine run? (Note/hint: see question 2)

Programmable refers to architectures which heavily and rapidly reuse a single piece of active circuitry for many different functions. The canonical example of a programmable device is a processor which may perform a different instruction on its ALU on every cycle.

Configurable refers to architectures where the active circuitry can perform any of a number of different operations, but the function cannot be changed from cycle to cycle. FPGAs are our canonical example of a configurable device. Once the instruction has been configured into the device, it is not changed during an operational epoch.

(https://ic.ese.upenn.edu/transit/rcgp/chapter1.4.2.html)

F-cam is more of a configurable architecture since it is not changed on a cycle-by-cycle basis like a programmable device. Instead, the system is configured to perform a specific set of operations.

It runs C++ to create custom applications and algorithms for capturing and processing images. The program operations can control the camera hardware, like sensors, devices and many other components to custom image processing tasks.

It's always good to establish the scope of what a system is trying to do. In this case, how would you characterize the particular type of computational photography algorithms that F-cam seeks to support/facilitate/enable?

The metering algorithm operates on the image histogram to maximize overall brightness while minimizing the number of oversaturated pixels.

The autofocus algorithm sweeps the lens from far focus to near focus and inspects the sharpness map.

The image processing algorithm allows programmers to use any image processing library they like for analysis and transformation.

All these algorithms can improve the quality of photographs captured by F-cam. They also make it more flexible by providing flexible and customizable algorithms for image processing.

Part 3: [Optional] Questions I'd like to have specifically addressed via in class. (We also encourage you to just post these questions on Ed immediately so anyone can answer!)