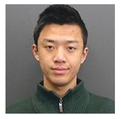
A System for Capturing Datasets for Mobile Image Matching


Nicholas Mosier '20


Tommaso Monaco '20


Roger Dai '20


Daniel Scharstein

ABSTRACT

This project aims to produce datasets of scenes imaged with mobile phones, consisting of images, videos, and highly-accurate 3D scene geometry. These datasets will be used in the next generation of the Middlebury stereo vision benchmarks to evaluate 3D depth maps computed from smartphone images.

Stereo vision is a method for recovering depth information from 3D scenes using images taken from slightly different positions, similar to the views observed by a human's two eyes. Stereo matching, the process of finding pixel correspondences between the images, is a difficult problem.

Previous versions of the Middlebury stereo vision benchmark datasets have used high-resolution professional cameras mounted on a tripod. This project takes a new approach: a smartphone is mounted on a robot arm, and all data is captured using a native app and then shipped to a computer for processing. The MobileLighting control program, written in Swift and C++ for macOS, coordinates the entire data collection process, including wireless image capture on an iPhone, robot arm movement, projection of structured lighting, camera calibration, and image processing. The MobileLighting system uses structured lighting to encode precise position maps of the scene, which are then processed using a refined disparity matching pipeline adapted from the ActiveLighting system used in previous datasets.

The research utilizes a UR5 robot arm that moves between different reference points. The robot aims to replicate actual human arm movements captured with an Xsens motion capture system. The recorded trajectory is smoothed before feeding it to the robot arm. The smartphone is mounted on the robot arm and takes a video along the trajectory. Separate frames are extracted from the video and then used in the image processing pipeline.

The system is calibrated using a custom version of AruCo barcode patterns instead of traditional chessboard calibration patterns. This allows accurate calibration results from multiple viewpoints even if the pattern is partially occluded in some of the views.

The long-term goal of the research is to produce the next generation of datasets and benchmarks for mobile image-matching applications.

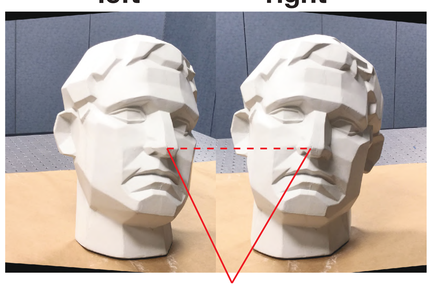
Goals and Motivation

- Promising applications of stereo image matching on mobile devices (e.g. VR, AR, e-commerce)
- Lack of high-quality "ground-truth" datasets for mobile devices
- Continue fundamental research on stereo matching and 3D modeling driven by real-world applicability
- Create system for capturing datasets on mobile devices

Stereo Image Matching

Stereo vision is the usage of images from two distinct viewpoints to extract 3D depth information of a scene. *Stereo image matching* is a method for recovering such depth information: each pixel in the left image is matched with a corresponding pixel in the right image. The disparities between corresponding pixels are inversely proportional to the 3D depth of the point in the scene.

left **right**



The MobileLighting System

The MobileLighting system coordinates all parts of dataset production, including the following:

- Captures "ground-truth" (GT) binary code images
- Projects structured light from multiple projectors
- Ambient image and video capture
- Camera calibration
- Robot arm movement
- Image processing (disparity matching, reprojection, etc.)

The system consists of two programs:

MobileLighting Control

- controls data acquisition and processing

MobileLighting Capture

- captures all data

Camera Calibration

Camera calibration encompasses the estimation of intrinsic and extrinsic parameters of the camera model.

Intrinsic Calibration

Intrinsic parameters encode the camera's manufactured characteristics (optical and geometric). They include

1. camera matrix:
   (a) focal lengths
   (b) principal points
2. distortion matrix:
   (a) radial distortion
   (b) tangential distortion

Extrinsic Calibration

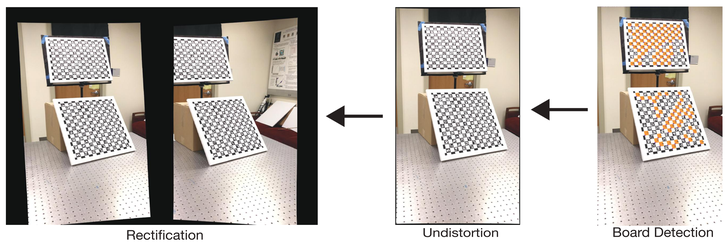
From intrinsic parameters, it is possible to compute the extrinsic parameters of a stereo setup. They include

1. translation matrix
2. rotation matrix

These define the relationship between the camera's reference frame and the world reference frame. After computing the extrinsic parameters, stereo rectification can be performed.

AruCo

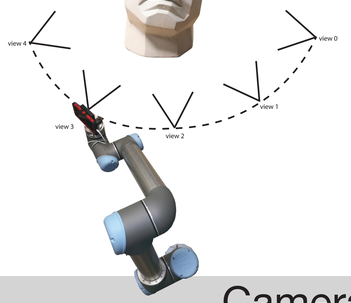
We employ a barcode pattern named AruCo. Our custom AruCo pattern works successfully even with partially occluded views. The entire calibration procedure has been rewritten using OpenCV3. A novel feature: the option to use multiple boards in the same view.

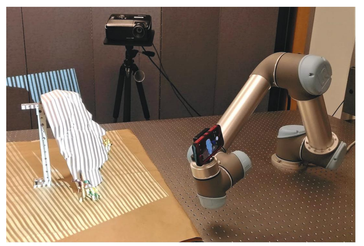


Dataset Acquisition

1. Calibration images with AruCo boards
2. Structured light: images of code patterns
3. Ambient photos (multiple exposures)
   (a) Normal lighting
   (b) With flashlight
   (c) With flash
4. Ambient video + IMU data*
   (a) Normal lighting
   (b) With blinking flashlight

** Inertial Measurement Unit data. Includes acceleration, attitude, and gyroscope samples.*





Camera Motion

The UR5 robot arm that we are using can be instructed to move to a certain position in two ways: joint positions and pose. The robot arm has six joints, therefore a position could be specified using an array of six angles, one for each joint. This gives a unique configuration of the robot arm. The pose, on the other hand, contains six numbers as well. The first three are the XYZ positions of the robot's tool head, in other words the tip of the arm. The next three are the rotation angles of the tool head around the XYZ axes. The first XYZ positions are mapped with the base frame of the robot arm as the reference frame, while the rotation uses the tool head frame. Our main method of communicating with the robot is through sockets, which allow us to send strings to the robot arm from a script on the computer. The robot arm runs its own programming language, which is UR script, uses that to interpret the received strings, and carries out the instructions.

We have investigated a several different options to plot the trajectory for the robot arm to move through. The easiest one is manually recording a fixed number of positions and have the robot arm move through them smoothly. A more advanced option is to have the robot arm mimic human movements as accurately as possible using motion capture, which is also our end goal. We used the Xsens motion capture kit to record the human arm movements and translated the data with MATLAB to generate the pose data mentioned earlier to send to the robot arm.

Image Processing Pipeline

New features

- 3 decoding interpolation modes:
  - bilinear interpolation
  - diagonal interpolation
  - planar interpolation
- Rectification of decoded images
  - computes intermediate results using linear and nearest-neighbor refinement
  - merges intermediate results into final result
  - minimizes information loss

Legend