Supplemental Document: Software Documentation

Low-Cost SPAD Sensing for Non-Line-Of-Sight Tracking, Material Classification and Depth Imaging

We provide code for every application described in the main paper so that the reader may reproduce our results or use the VL53L1X sensor for their own project. For technical reasons, the current version of the code contains Python and Matlab scripts. We plan to convert all code to Python in a later revision. The usage of the code is explained in the following sections. It works on Windows and Linux.

1 GENERAL PROCEDURE

For all our applications, we use the STMicroelectronics P-NUCLEO-53L1A1 evaluation kit which consists of the STM32F401RE nucleo board, a VL53L1X expansion board (X-NUCLEO-53L1A1) and two VL53L1X breakout 'satellite' boards. For all applications, please follow the steps below.

1. Connect the nucleo board to your computer using a USB cable. The device should appear as a volume called NODE_F401RE.
2. Copy the firmware you want to use from the `firmware` directory onto the volume. The device will automatically reboot and start running the firmware. All of our firmwares run the *right* (marked on the board) satellite sensor. The left satellite does not need to but can be connected.
3. Determine the serial port name and number of the device on your system (e.g. `COM3` on Windows or `ttyACM0` on Linux). You will need to adjust this name in the scripts that access the device via the serial port.

2 MATERIAL CLASSIFICATION

We provide code to acquire custom training data for the classification of materials (`capture_training_data.m`), code to train the network on material data (`materialclassification_training.py`), and a script to test the trained network on different materials `live_evaluate_5materials.py`).

Capturing training data:

- Copy `MatClass_trainingDataAc.bin` to the nucleo board.
- Adjust your serial port name in `capture_training_data.m` and run it in Matlab while holding the sensor to the material. It will start the data acquisition and tell you when to move the position of the sensor. It captures 40 different points with 25 histograms each.

Training the network:

- Run `materialclassification_training.py`. It uses our provided material data by default.

Live evaluation of material classification:

- Copy `MatClass_liveData.bin` to the nucleo board.
- Run `live_evaluate_5materials.py` and hold the sensor to a material. The data is evaluated by the network and the predicted material is displayed in a python console.

3 DEPTH IMAGING

This application requires scanning the scene with a galvanometer mirror system. Our system consists of an off-the-shelf galvanometer mirror system and an Arduino. Instructions on how to set up such a system can, for instance, be found in [DeltaFlo 2016]. Note that the mirrors have to be reflective at 940 nm. Since the standard mirrors in our system were transparent to that wavelength, we equipped it with additional, bigger hot mirrors, and 3d-printed a custom mount for a fitting adjustment.

Capturing depth data:

- Place two plano-convex lenses in front of the VL53L1X light source and sensor to collimate the illumination and the sensor's field of view.
- Align the sensor with the mirror system.
- Transfer the depth imaging `hex` file to your Arduino.
- Adjust the Arduino's and nucleo board's serial port name in `depth_data_acquisition.m` and run the script to capture a depth image with 128×128 mirror positions.

Create depth images from captured depth data:

- Run `create_depth_images.m` in Matlab.
- For sharper results, you may want to measure your own system's point spread function using a small (~2 mm × 2 mm) retroreflective patch on a background that is as non-reflective as possible at 920 nm.

4 NON-LINE-OF-SIGHT TRACKING

Code for all four methods described in the main paper are provided along with data and training checkpoints for both targets and both setups (with and without mirrors). The desired configuration can be set in the Python script file `scriptloader.py` which will then run the corresponding python script with the appropriate parameters.

Capturing NLOS data:

- Copy `NLOS_withoutMirrors_trainingDataAc.bin` or `NLOS_withMirrors.bin` to the nucleo board, depending on whether your setup does or doesn't employ a mirror setup.
- If you use mirrors, transfer the NLOS `hex` file to your Arduino.
- Run `NLOS_data_acquisition_nomirrors.m` or `NLOS_data_acquisition_mirrors.m`, depending on your setup. The script will save individual measurement files.
- To combine all measurements into one file to be read by a reconstruction script, and calculate further quantities that may be needed, run `process_measurements.m`.

REFERENCES

DeltaFlo. 2016(?). Arduino Laser Show With Real Galvos. (2016(?)). https://web.archive.org/web/20201111233107/https://www.instructables.com/Arduino-Laser-Show-With-Real-Galvos/