**Mapping Steering Mirror Coordinates to Target Plane Coordinates**

To direct the laser beam MR-E-2 beam steering mirror is used. Laser beam is produced by a laser which is stationary. Laser is positioned to hit the steering mirror in the center. After the beam hits the mirror, it is directed to required position by adjusting the position of the mirror.

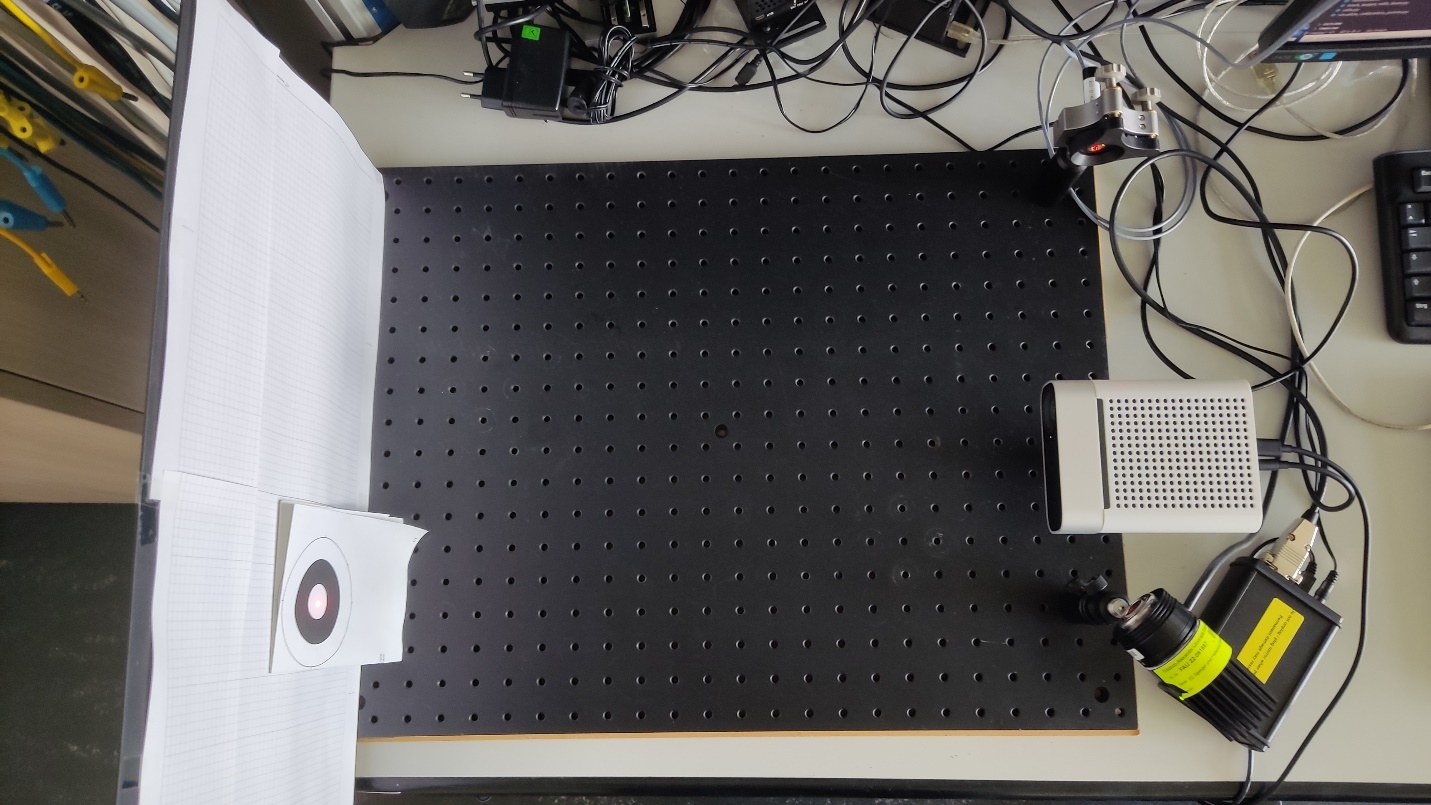


Figure : Laser, depth camera, and beam steering mirror setup

Using the mirror to steer the laser beam requires a mapping from 3D world coordinate system to mirror coordinate system. The mapping calculates the required rotation of the mirror around its axes to direct the laser beam to intended 3D position. Mirror coordinate system is defined in [1] as in following figure.

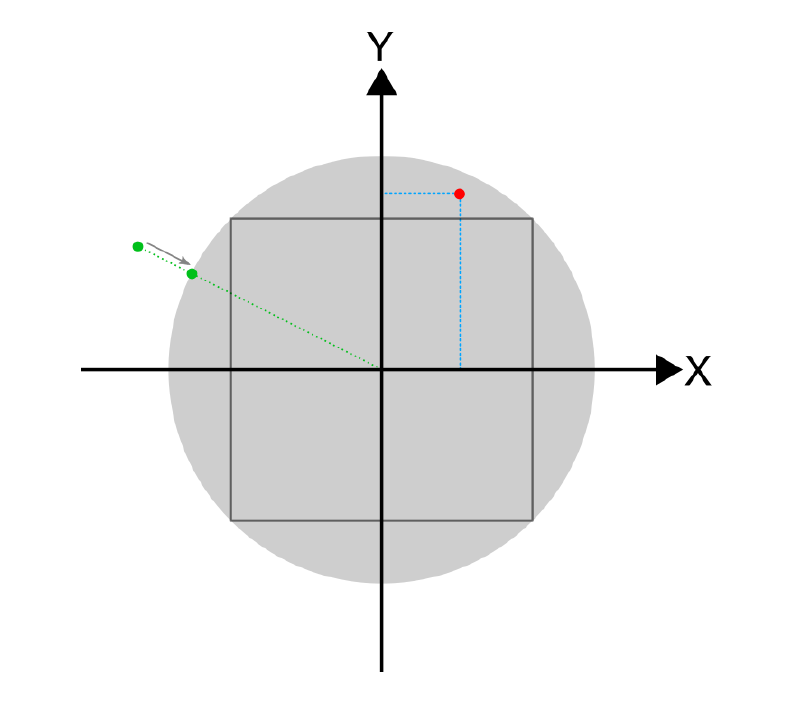


Figure : Internal mirror coordinate system

The coordinate system is a Cartesian coordinate system with X and Y axes. Rotation of the mirror around its horizontal and vertical axes are expressed as x and y values. The range of both axes are [-1, 1] interval. Mirror has maximum deflection angle of 25°. Θ is the angle between incoming and reflected beam. Θ = +50° corresponds to +1 and Θ = -50° corresponds to -1. X and Y values are required to be inside a unit circle in order to be a valid point accessible by the mirror.

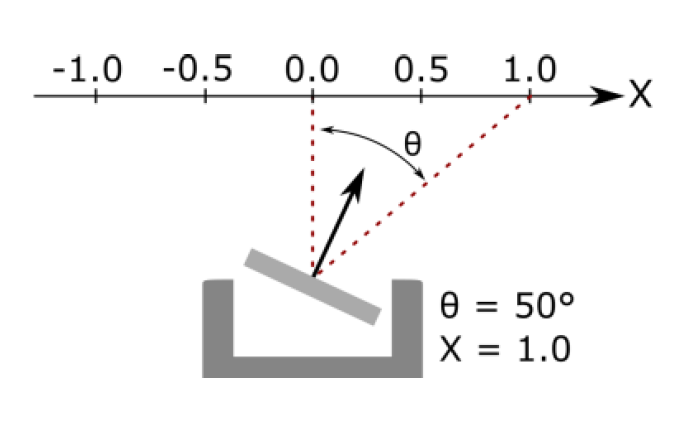


Figure : Relation between rotation of the mirror (Θ) and the mirror coordinate (x) in X-axis

****

Figure : Mirror coordinate system [1]

Variables used in Figure 4 and their descriptions:

P0: incoming beam origin

n0: incoming beam direction (unit vector)

M: mirror center

C: center of rotation

d: distance between M and C

nm: mirror normal vector

P1: reflected beam origin

n1: reflected beam direction (unit vector)

D: distance between mirror center(C) and target plane

nt: target plane normal vector

P2: the point reflected beam hits the target plane

alpha: rotation degree of the target plane with respect to steering mirror

In the system setup used in experiments following configuration is used:

* 𝑃1 = 𝑀 = 𝐶
* 𝑂 = 𝑃1 = 𝑀 = 𝐶
* d=0
* Incoming ray is coming in y-z plane.

The conversion of coordinates on target plane (x\_t, y\_t) to mirror coordinates (x, y) is performed in two main steps. Firstly, the mirror normal (n\_m) is calculated based on position of the laser, position of the target plane and (x\_t, y\_t). In the next step, corresponding mirror coordinates (x, y) is calculated using n\_m.

1) Computation of mirror normal vector (n\_m)

There are 2 different frames of reference I and T. Reference frame I is centered around O. Its z-axis is aligned with -n\_m direction and its y-axis is aligned so that incoming laser beam propagates through yz plane. Reference frame T is centered around T. Its z-axis is aligned with n\_t direction and its y-axis is aligned so that reflected laser beam propagates through yz plane. The transformation between frame of references is done with orthogonal transformation matrices A\_IT and A\_TI.

T\_n\_t = [0, 0, 1].T

T\_r\_OT = [0,0,-D].T

T\_r\_TP\_2 = [x\_t, y\_t, 0].T

I\_n\_0 = normalize([0,-1,1].T)

A\_IT = [[ 1, 0 ,0],

[ 0, cos(alpha), -sin(alpha)],

[ 0, sin(alpha), cos(alpha)]]

Mirror normal vector is calculated with following steps:

* I\_n\_t = A\_IT@ T\_n\_t
* I\_r\_OT = A\_IT @ T\_r\_OT
* I\_r\_TP\_2 = A\_IT @ T\_r\_TP\_2
* I\_r\_OP\_2 = r\_OT + I\_r\_TP\_2
* I\_n\_1 = normalize(r\_OP\_2)
* I\_n\_m = normalize(n\_1-n0)

2) Computation of mirror coordinates (x, y)

I\_r\_C = [0, 0, d].T

I\_n\_0 = normalize([0,0,1].T)

I\_r\_OP\_0 = [0,0, -1].T

T\_n\_t = [0,0,1].T

T\_ r\_OT = [0,0,-D].T

A\_IT = [[ 1, 0 ,0],

[ 0, 1, 0],

[ 0,0 , 1]]

I\_n\_0 = [0, 0, 1].T

* t\_1 = ((I\_r\_C – I\_r\_OP\_0) @ I\_n\_m + d) / (I\_n\_0 @ I\_n\_m)
* I\_r\_OP\_1 = I\_r\_OP\_0 + t\_1 \* I\_n\_0
* I\_n\_1 = I\_n\_0 - 2\*(I\_n\_0 @ I\_n\_m) @ I\_n\_m
* t\_2 = ((T\_r\_OT – I\_r\_OP\_1)@ T\_n\_t) / (I\_n\_1 @ T\_ n\_t)
* I\_r\_OP\_2 = I\_r\_OP\_1 + t\_2 \* I\_n\_1
* x = I\_r\_OP\_2[0]/( D\*tan(50))
* y = I\_r\_OP\_2[1]/ (D\*tan(50))

**Position Error Measurement System**

Measurements are done with a 2-dimensional servo motor system with a PM400 power meter attached.

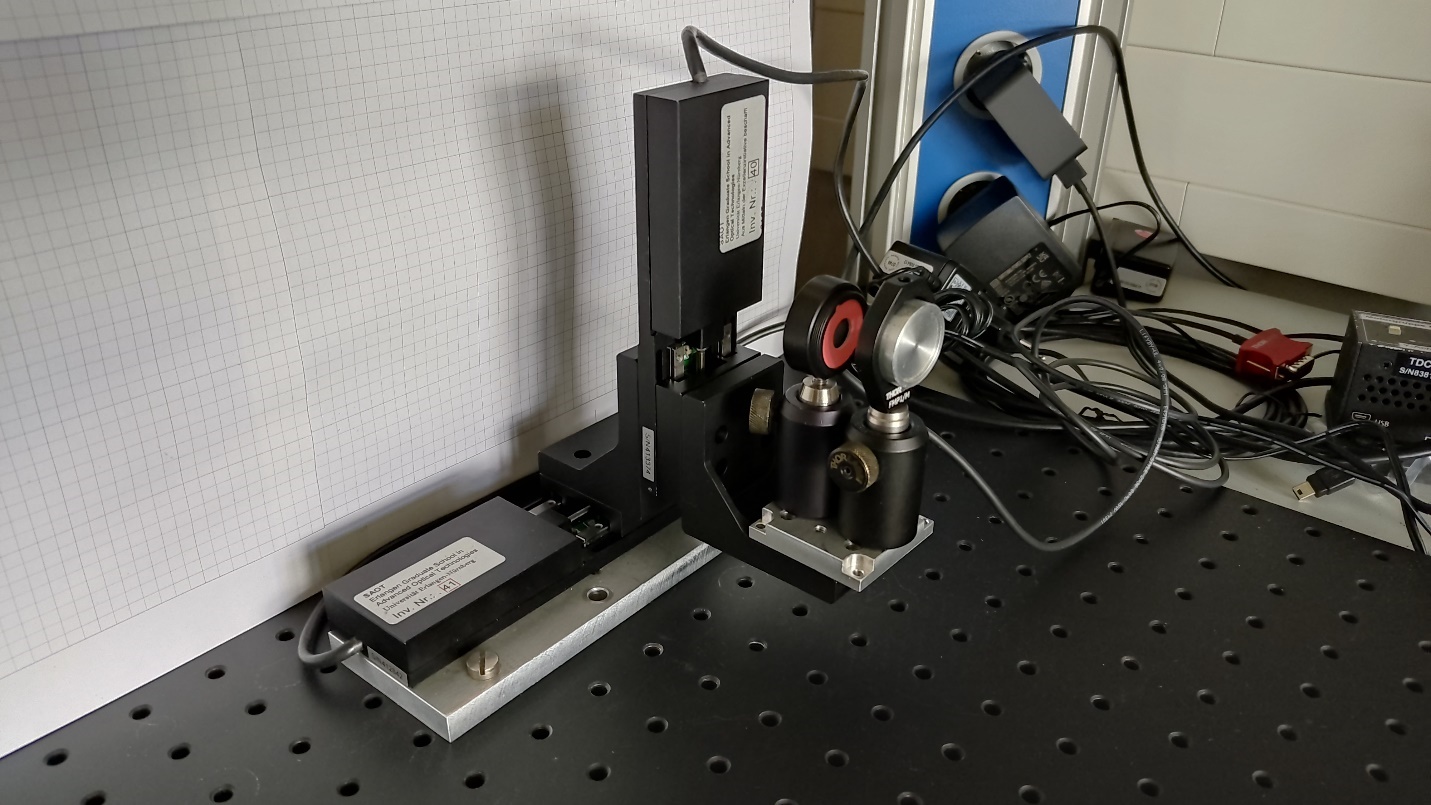


Figure : Position error test setup

Measurements are performed by pointing the laser to a specified target position and then sweeping the power meter in a linear trajectory. Power recordings are recorded with recording times and plotted to find the instance with maximum power. The point with maximum power gives the center position of the laser. Power measurements are performed by a Python script which has a loop with a period of around 0.01s. This sampling period is not constant throughout the measurements and might deviate from 0.01s. For this reason, after the measurement, cubic interpolation is performed to get a uniformly sampled signal.

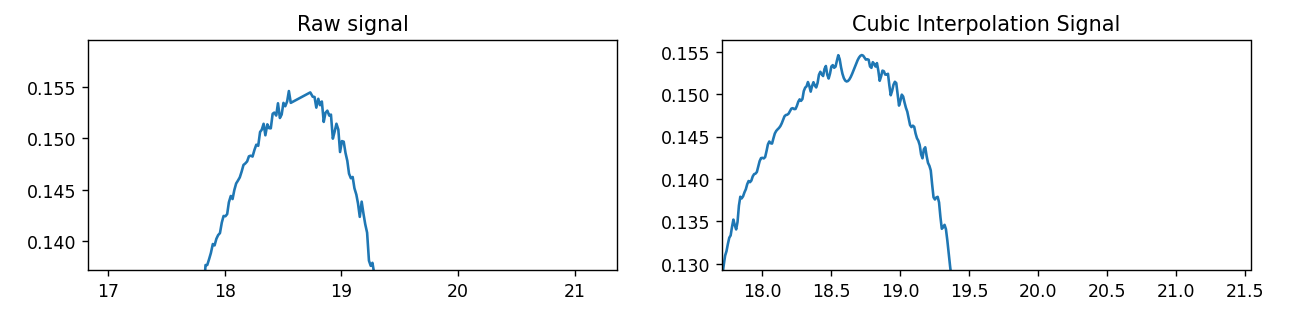
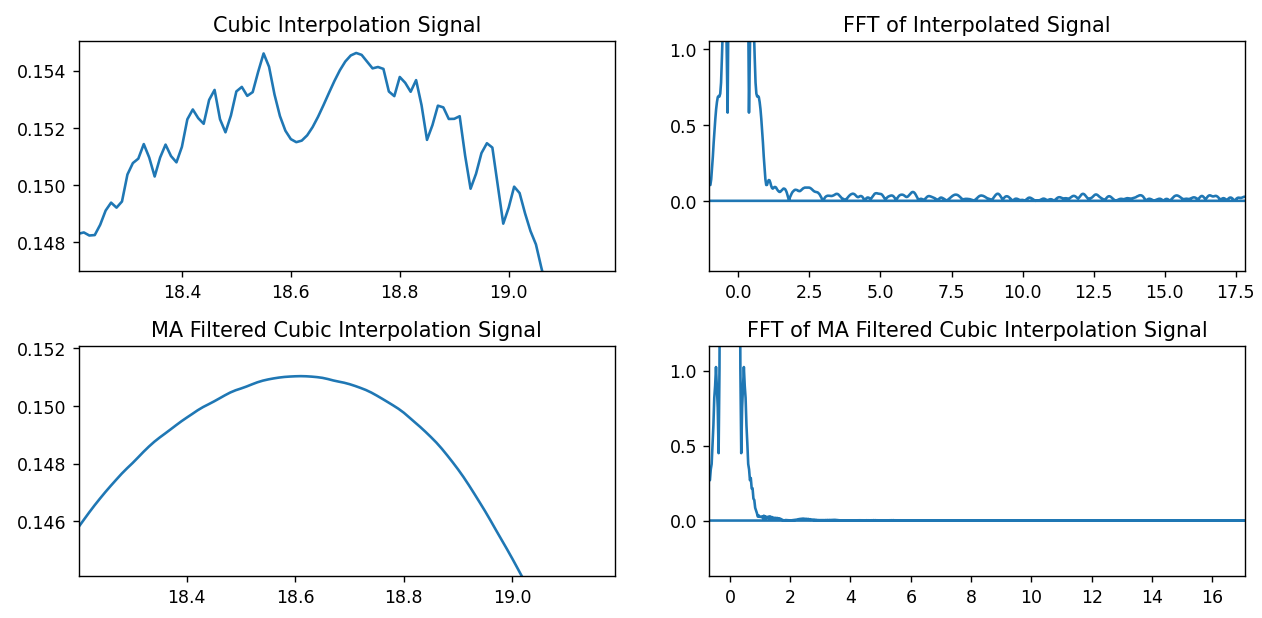
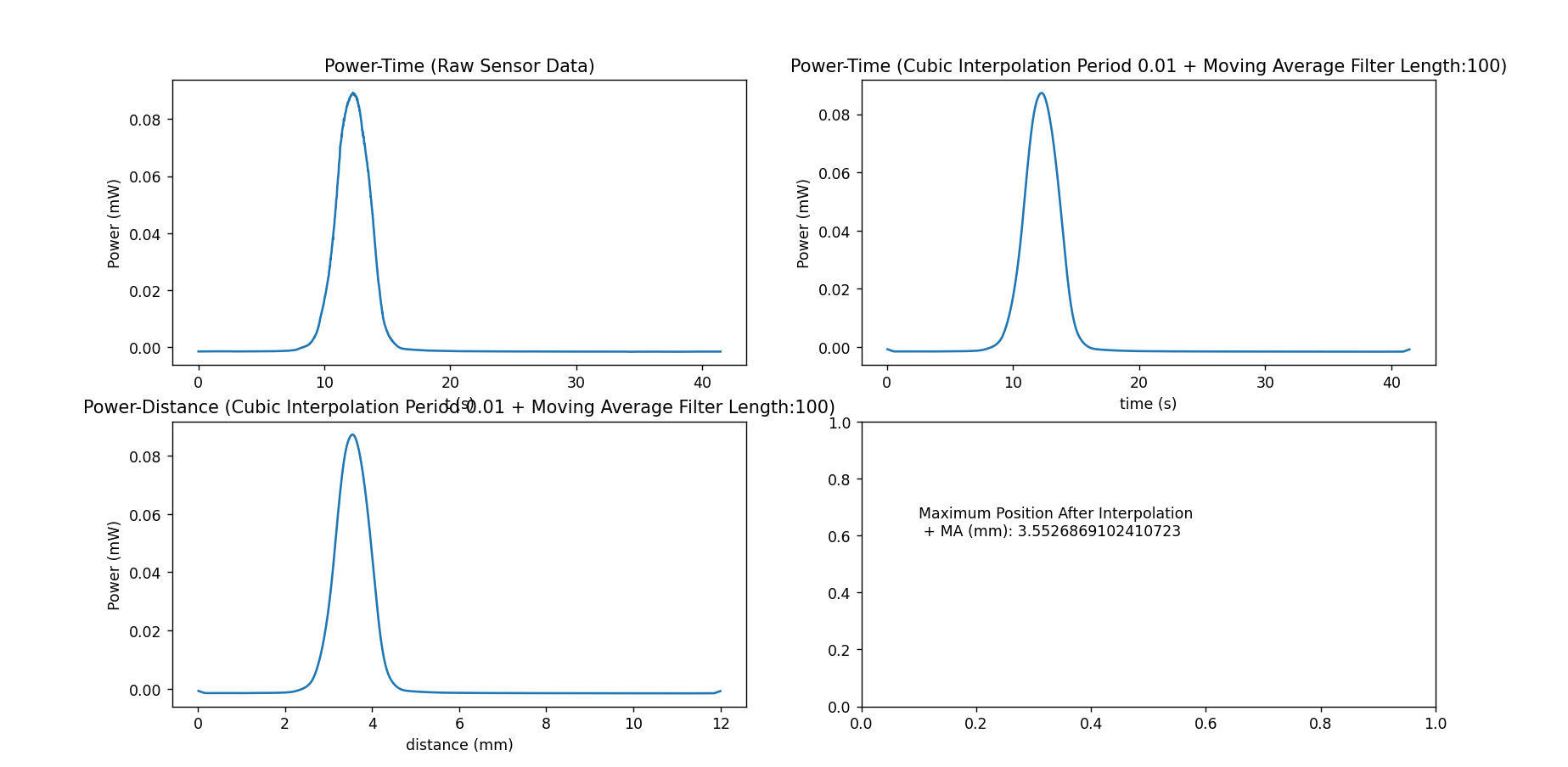
****

Figure : Power(mW) - Position(mm) graphs for raw signal and cubic interpolation signal

Interpolation solves the nonuniform sampling problem. Obtained signal still has noise in it which might alter the maximum position. To remove the noise a low pass filter in the form of a moving average filter is applied. The figure below depicts the signal before and after the low pass filter. High-frequency noise components are removed from the signal while keeping the original signal mostly intact. This operation produced a smoother signal which is more suitable for peak finding operation.

****The recorded power signal is plotted with respect to distance in order to find the position of the laser’s center. This method gives the position of the laser relative to the initial position of the servo motor. The figure below shows the power levels with respect to time and position. Position with maximum power is written in the fourth graph in millimeters.



Measurements are performed with different parameters such as the distance between the mirror and target plane, different sweep locations.

**Horizontal Measurements D=410mm (actual distance 425mm)**

Table : Distance between mirror and target plane: 410mm. Mirror input y coordinate: -5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input x (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | -5 | 2.322 | -3.096 |
| -2.5 | -5 | 2.853 | -2.565 |
| -2 | -5 | 3.372 | -2.046 |
| -1.5 | -5 | 3.868 | -1.550 |
| -1 | -5 | 4.392 | -1.026 |
| -0.5 | -5 | 4.905 | -0.513 |
| 0 | -5 | 5.418 | 0 |
| 0.5 | -5 | 5.921 | 0.503 |
| 1 | -5 | 6.446 | 1.028 |
| 1.5 | -5 | 6.923 | 1.505 |
| 2 | -5 | 7.427 | 2.009 |
| 2.5 | -5 | 7.937 | 2.519 |
| 3 | -5 | 8.452 | 3.034 |

Table : Distance between mirror and target plane: 410mm. Mirror input y coordinate: 0mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 0 | 2.278 | -3.089 |
| -2.5 | 0 | 2.792 | -2.575 |
| -2 | 0 | 3.311 | -2.056 |
| -1.5 | 0 | 3.829 | -1.538 |
| -1 | 0 | 4.327 | -1.040 |
| -0.5 | 0 | 4.860 | -0.507 |
| 0 | 0 | 5.367 | 0 |
| 0.5 | 0 | 5.878 | 0.511 |
| 1 | 0 | 6.379 | 1.012 |
| 1.5 | 0 | 6.880 | 1.513 |
| 2 | 0 | 7.392 | 2.025 |
| 2.5 | 0 | 7.880 | 2.513 |
| 3 | 0 | 8.390 | 3.023 |

Table : Distance between mirror and target plane: 410mm. Mirror input y coordinate: 5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 5 | 2.391 | -3.097 |
| -2.5 | 5 | 2.909 | -2.579 |
| -2 | 5 | 3.424 | -2.064 |
| -1.5 | 5 | 3.949 | -1.539 |
| -1 | 5 | 4.468 | -1.020 |
| -0.5 | 5 | 4.974 | -0.514 |
| 0 | 5 | 5.488 | 0 |
| 0.5 | 5 | 6.013 | 0.525 |
| 1 | 5 | 6.534 | 1.046 |
| 1.5 | 5 | 7.042 | 1.554 |
| 2 | 5 | 7.547 | 2.059 |
| 2.5 | 5 | 8.052 | 2.564 |
| 3 | 5 | 8.561 | 3.073 |

**Horizontal Measurements D=425mm (actual distance 425mm)**

Table : Distance between mirror and target plane: 425mm. Mirror input y coordinate: -5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input x (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | -5 | 6.723 | -2.999 |
| -2.5 | -5 | 7.225 | -2.497 |
| -2 | -5 | 7.727 | -1.995 |
| -1.5 | -5 | 8.218 | -1.504 |
| -1 | -5 | 8.716 | -1.007 |
| -0.5 | -5 | 9.228 | -0.494 |
| 0 | -5 | 9.722 | 0.0 |
| 0.5 | -5 | 10.212 | 0.49 |
| 1 | -5 | 10.695 | 0.973 |
| 1.5 | -5 | 11.191 | 1.469 |
| 2 | -5 | 11.673 | 1.951 |
| 2.5 | -5 | 12.151 | 2.429 |
| 3 | -5 | 12.653 | 2.931 |

Table : Distance between mirror and target plane: 425mm. Mirror input y coordinate: 0mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 0 | 6.788 | -2.956 |
| -2.5 | 0 | 7.275 | -2.469 |
| -2 | 0 | 7.777 | -1.967 |
| -1.5 | 0 | 8.289 | -1.455 |
| -1 | 0 | 8.758 | -0.986 |
| -0.5 | 0 | 9.248 | -0.496 |
| 0 | 0 | 9.744 | 0.0 |
| 0.5 | 0 | 10.225 | 0.481 |
| 1 | 0 | 10.711 | 0.967 |
| 1.5 | 0 | 11.207 | 1.463 |
| 2 | 0 | 11.666 | 1.922 |
| 2.5 | 0 | 12.156 | 2.412 |
| 3 | 0 | 12.654 | 2.91 |

Table : Distance between mirror and target plane: 425mm. Mirror input y coordinate: 5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 5 | 6.849 | -2.978 |
| -2.5 | 5 | 7.364 | -2.463 |
| -2 | 5 | 7.867 | -1.959 |
| -1.5 | 5 | 8.359 | -1.468 |
| -1 | 5 | 8.843 | -0.984 |
| -0.5 | 5 | 9.34 | -0.487 |
| 0 | 5 | 9.826 | 0.0 |
| 0.5 | 5 | 10.308 | 0.481 |
| 1 | 5 | 10.824 | 0.998 |
| 1.5 | 5 | 11.304 | 1.478 |
| 2 | 5 | 11.793 | 1.967 |
| 2.5 | 5 | 12.279 | 2.453 |
| 3 | 5 | 12.781 | 2.954 |

**Horizontal Measurements D=210mm (actual distance 225mm)**

Table : Distance between mirror and target plane: 210mm. Mirror input y coordinate: -5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | -5 | 3.387 | -3.230 |
| -2.5 | -5 | 3.913 | -2.704 |
| -2 | -5 | 4.454 | -2.163 |
| -1.5 | -5 | 5.000 | -1.617 |
| -1 | -5 | 5.533 | -1.084 |
| -0.5 | -5 | 6.082 | -0.535 |
| 0 | -5 | 6.617 | 0 |
| 0.5 | -5 | 7.170 | 0.553 |
| 1 | -5 | 7.700 | 1.083 |
| 1.5 | -5 | 8.233 | 1.616 |
| 2 | -5 | 8.765 | 2.148 |
| 2.5 | -5 | 9.296 | 2.679 |
| 3 | -5 | 9.843 | 3.226 |

Table : Distance between mirror and target plane: 210mm. Mirror input y coordinate: 0mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 0 | 3.359 | -3.204 |
| -2.5 | 0 | 3.888 | -2.675 |
| -2 | 0 | 4.416 | -2.147 |
| -1.5 | 0 | 4.954 | -1.609 |
| -1 | 0 | 5.490 | -1.073 |
| -0.5 | 0 | 6.026 | -0.537 |
| 0 | 0 | 6.563 | 0 |
| 0.5 | 0 | 7.104 | 0.541 |
| 1 | 0 | 7.647 | 1.084 |
| 1.5 | 0 | 8.179 | 1.616 |
| 2 | 0 | 8.692 | 2.129 |
| 2.5 | 0 | 9.221 | 2.658 |
| 3 | 0 | 9.754 | 3.191 |

Table : Distance between mirror and target plane: 210mm. Mirror input y coordinate: 5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 5 | 3.553 | -3.189 |
| -2.5 | 5 | 4.077 | -2.665 |
| -2 | 5 | 4.604 | -2.138 |
| -1.5 | 5 | 5.158 | -1.584 |
| -1 | 5 | 5.679 | -1.063 |
| -0.5 | 5 | 6.212 | -0.529 |
| 0 | 5 | 6.742 | 0.0 |
| 0.5 | 5 | 7.27 | 0.528 |
| 1 | 5 | 7.807 | 1.065 |
| 1.5 | 5 | 8.343 | 1.601 |
| 2 | 5 | 8.873 | 2.131 |
| 2.5 | 5 | 9.404 | 2.662 |
| 3 | 5 | 9.934 | 3.192 |

**Horizontal Measurements D=225mm (actual distance 225mm)**

Table : Distance between mirror and target plane: 225mm. Mirror input y coordinate: -5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | -5 | 4.985 | -3.044 |
| -2.5 | -5 | 5.503 | -2.526 |
| -2 | -5 | 6.007 | -2.023 |
| -1.5 | -5 | 6.505 | -1.524 |
| -1 | -5 | 7.034 | -0.995 |
| -0.5 | -5 | 7.536 | -0.494 |
| 0 | -5 | 8.029 | 0.0 |
| 0.5 | -5 | 8.539 | 0.509 |
| 1 | -5 | 9.045 | 1.016 |
| 1.5 | -5 | 9.546 | 1.517 |
| 2 | -5 | 10.045 | 2.015 |
| 2.5 | -5 | 10.529 | 2.499 |
| 3 | -5 | 11.009 | 2.98 |

Table : Distance between mirror and target plane: 225mm. Mirror input y coordinate: 0mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 0 | 4.989 | -2.992 |
| -2.5 | 0 | 5.493 | -2.488 |
| -2 | 0 | 5.999 | -1.982 |
| -1.5 | 0 | 6.49 | -1.491 |
| -1 | 0 | 6.998 | -0.984 |
| -0.5 | 0 | 7.459 | -0.523 |
| 0 | 0 | 7.981 | 0.0 |
| 0.5 | 0 | 8.461 | 0.479 |
| 1 | 0 | 8.952 | 0.971 |
| 1.5 | 0 | 9.485 | 1.503 |
| 2 | 0 | 9.976 | 1.994 |
| 2.5 | 0 | 10.464 | 2.483 |
| 3 | 0 | 10.956 | 2.975 |

Table : Distance between mirror and target plane: 225mm. Mirror input y coordinate: 5mm

|  |  |  |  |
| --- | --- | --- | --- |
| **Mirror Input (mm)** | **Mirror Input y (mm)** | **Peak Power Position (mm)** | **Relative Distance (mm)** |
| -3 | 5 | 5.156 | -2.982 |
| -2.5 | 5 | 5.664 | -2.475 |
| -2 | 5 | 6.153 | -1.986 |
| -1.5 | 5 | 6.649 | -1.49 |
| -1 | 5 | 7.154 | -0.985 |
| -0.5 | 5 | 7.64 | -0.498 |
| 0 | 5 | 8.138 | 0.0 |
| 0.5 | 5 | 8.634 | 0.496 |
| 1 | 5 | 9.126 | 0.988 |
| 1.5 | 5 | 9.635 | 1.497 |
| 2 | 5 | 10.126 | 1.988 |
| 2.5 | 5 | 10.609 | 2.47 |
| 3 | 5 | 11.103 | 2.965 |

**Mapping Depth Camera Coordinate System to Steering Mirror Coordinate System**

The coordinate systems that are used in the steering mirror and the depth camera don’t coincide. They are rotated and translated versions of each other. A point in camera coordinate system, c, can be transformed into a point in mirror coordinate system, m, with the following mapping:

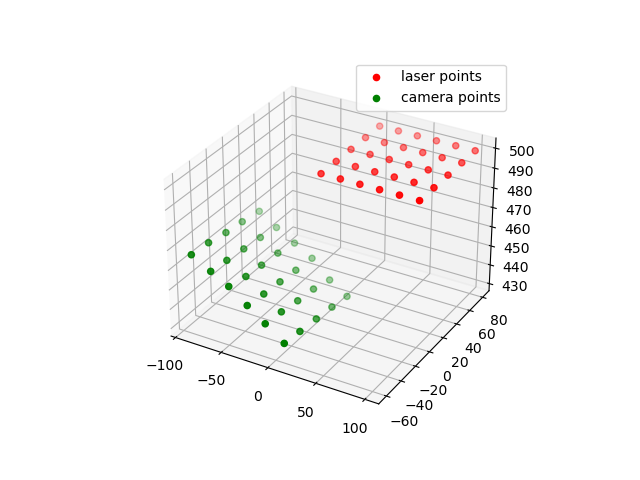


Figure : Recording of the same points in steering mirror and depth camera coordinate systems

R3x3 is an orthogonal rotation matrix and t3x1 is a translation vector. Optimal and t are found by the following algorithm [2] [3] [4]:

To find the rotation matrix R

To find the translation vector t

Optimal transformation maps each point in camera coordinate system to a point in mirror coordinate system. After the mapping mirror points and camera points are almost aligned as shown in Figure 8. With this mapping, any point recorded by depth camera can be transformed into a point which can be used by the mirror controller.

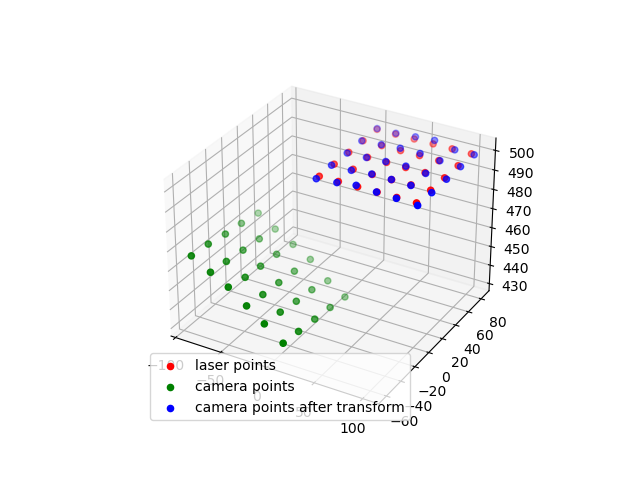


Figure : Camera points after the transformation

**Calibration Procedure**

Calibration of the system is done in order to coordinate depth camera and mirror together. Both steering mirror and depth camera have their own coordinate systems. The relationship between these coordinate systems is unknown which makes them unable to perform tasks together. The main objective of the calibration is to find the rotation and translation of the coordinate systems relative to each other. For this purpose, the optimal R and t finding algorithm discussed in previous chapter is used.

Calibration procedure is responsible for obtaining point pairs consisting of a point in depth camera coordinate system and its corresponding point in steering mirror coordinate system. After the collection of the dataset, the mapping is computed and saved for later usage. Point pair collection operation depends on the type of laser used in the system. For easier testing a visible laser is used. However, in the end system an IR laser is used. Using IR laser restricts the usage of camera since it is not visible by camera sensor. IR laser requires IR light intensity sensors to detect and a different calibration procedure than visible laser.

**1) Visible Laser**

The laser is pointed at 30 different positions in a plane 560 mm away from the mirror center. These points are recorded as mirror points . Each laser position is extracted from the color images by using Hough Circle Detection algorithm. Once 2D pixel coordinates are found they are converted to 3D coordinates by Kinect Azure SDK’s k4a::calibration::convert\_2d\_to\_3d [5] function. Extracted 3D depth camera points are named . Points and are formed into matrices with shape 3xN. Finally, algorithm 1 is used to find R and t matrices.

----Add picture of a laser point in calibration surface

**Add Calibration error table**

**2) IR Laser**

IR laser requires different calibration method from the visible laser. Point patching operation is done via laser intensity sensors. The sensor setup used to detect positions is in figure below. 3 sensors generate analog output signals based on the intensity of light hitting them. The analog signal is captured by ADCs on a Raspberry Pi Pico board and transmitted to host computer via USB.

--add sensor picture

While the sensor is recording the signal intensity, laser scans the area near the sensor position based on an initial sensor position estimate in terms of a 3D coordinate. Each signal point is plotted with respect to position of the laser at the time of recording. As a result, an intensity profile is generated. The maximum intensity level of this profile is chosen as the sensor position. This process is performed by all three sensors, and 3 different images are generated. In each image a peak intensity is observed, and it is marked as that specific sensor’s position.

--Add sensor intensity images

3 3D coordinates generated are not exact coordinates due to the initial position estimate. The x and y positions determined by scanning the laser are valid if and only if the initial distance between mirror center and target plane is correct. Since this won’t be the case for most cases, found points should be used to calculate the real distance between mirror center and target plane.

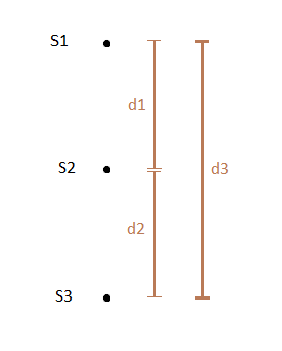
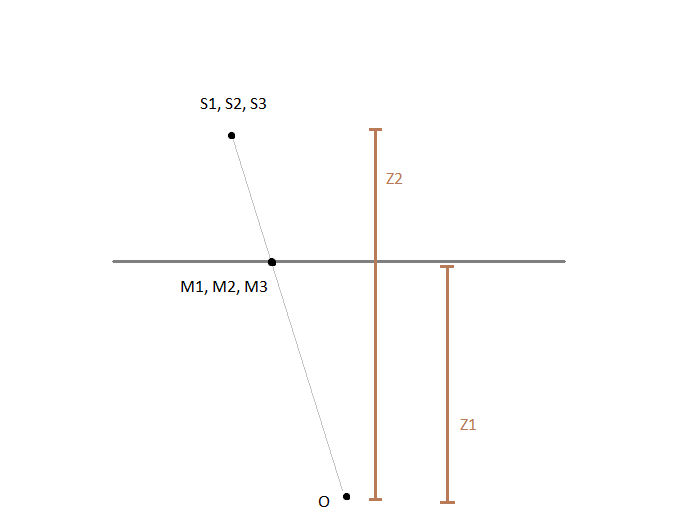
****

Figure : Sensor positions (S1, S2, S3), detected sensor positions (M1, M2, M3) and steering mirror position (O) from camera view (left) and top view (right)

To calculate the distance between mirror center and target plane, the geometry of the sensors is used. Sensors are placed on top of each other as in Figure 9 and the plate holding the sensors is placed perpendicular to ground. This configuration places all the sensors at the same distance from mirror center in the z direction. Using this constraint, the distance between sensors and mirror center in z direction is calculated:

d1 = |m1\*z2/z1 – m2\*z2/z1|

z2 = d1\*z1 / |m1-m2|

d2 = |m2\*z2/z1 – m3\*z2/z1|

z2 = d2\*z1 / |m2-m3|

d3 = |m1\*z2/z1 – m3\*z2/z1|

z2 = d3\*z1 / |m1-m3|

With the z2 distance obtained, previous 3D coordinates are updated and they are saved as mirror points .

The next step is to find the sensor positions with the depth camera and to obtain depth camera points . To localize the sensors, the marker pattern in figure below is attached to the sensor plate. Each circular fiducial marker is localized using WHYCon algorithm (will be discussed later). Two unit vectors are extracted from the 3D coordinates of markers.

u\_l = B-A

u\_d = C-A

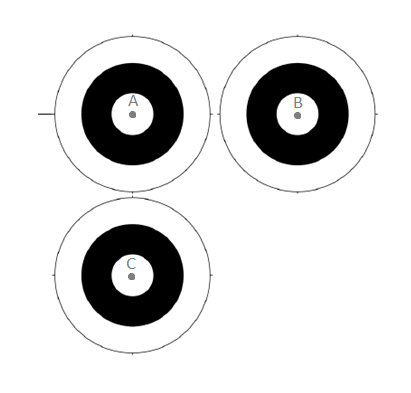
****

Figure : Localization marker pattern

Unit vectors u\_l and u\_d are used to locate the 3D position of the sensors. This is possible since markers and the marker pattern are on the same plane. Found points are recorded as depth camera points . Points and are formed into matrices with shape 3xN. Finally, algorithm 1 is used to find R and t matrices.

**Add Calibration error table**

**Detection Algorithms:**

Position estimation is the first step to tracking the position of the body. One of the most common approaches to this problem is usage of visual markers. A visual marker can be distinguished from the environment without too much difficulty due to its characteristics. For the project 3 different algorithms are tested: ArUco, Hough Circle Detector, and WHYCon. Detection algorithms are evaluated based on their processing times in order to choose the best option for the system.

**ArUco**

ArUco is a popular binary square fiducial marker used for absolute pose estimation. ArUCo markers are black square shapes with an inner grid structure to encode a binary code. The binary code is used to identify the markers and distinguish multiple markers inside the same camera view. Due to the encoding, Aruco markers are not easily detected from far distances. [6]

Figure : ArUco marker with ids 0 up to 3 [6]

**Hough Circle Detector**

**WHYCon**

WHYCon is an open-source marker-based localization system. It is highly computationally efficient. [7] WHYCon doesn’t provide ant identification information, its marker consists of black and white concentric circles. Due to its simplicity, it can be identified easily and it requires low computational resources. Durin implementation, only option that was able to run on a Windows machine was a C++ repository. To connect the algorithm to rest of the system, Python bindings were implemented using Pybind11.

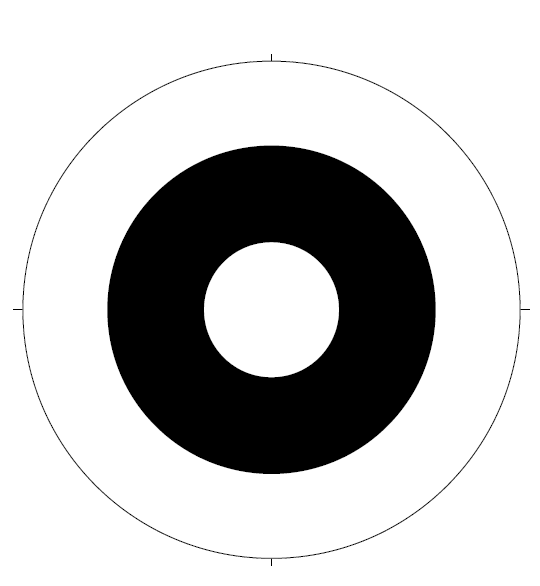
****

Figure : WHYCon marker [7]

**Algorithm Performance Comparison**

All of the algorithms are run on the same computer with Intel i7. Based on the runtimes WHYCon is the most performant algorithm. ArUco has similar performance with 4ms, but in case of movement and bad lighting conditions the runtime can increase up to 30ms. With a 30ms runtime the algorithm is not suitable for real time tracking. Hough Circle Detector also has good runtime, but WHYcon outperforms it. Due to its performance, WHYCon is used in the system.

|  |  |
| --- | --- |
| **Algorithm** | **Runtime (ms)** |
| ARuCO | 4-30 |
| WHYCon | 2-3 |
| Hough Circle Detector | 8-10 |

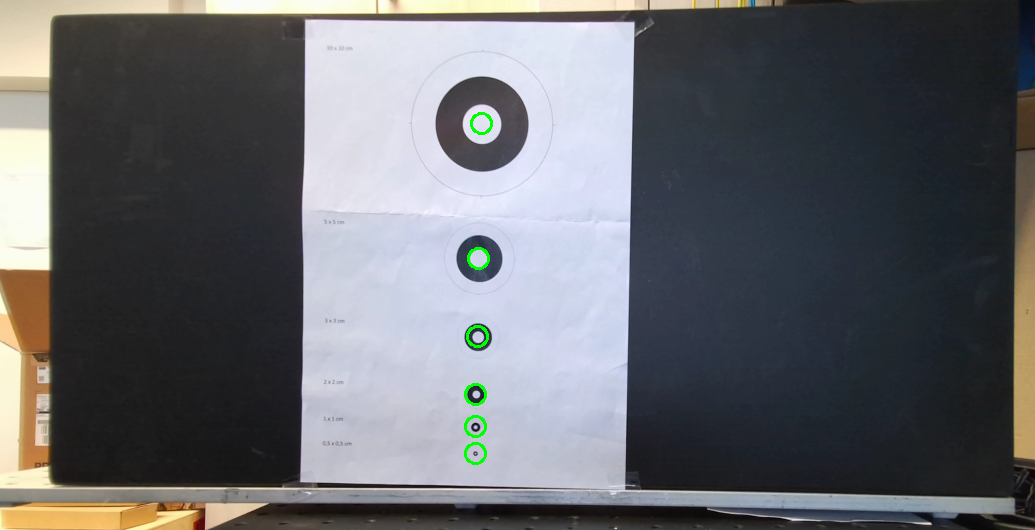
|  |  |
| --- | --- |
| **Parts of Code** | **Elapsed Time (ms)** |
| Capturing Color Image | 3 |
| Marker Detection | 2 |
| Transforming Depth Image to Color Camera´s Coordinate System | 15 |
| 3D to mirror coordinate conversion | 1 |
| Mirror Adjustment | 2 |
| **Total time** | **23** |

**Maximum distance of detection with different marker sizes:**

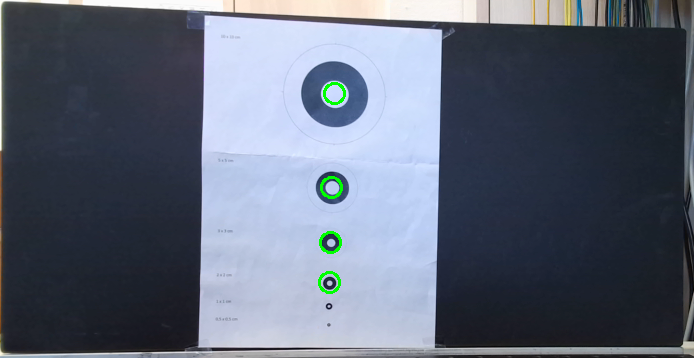
**-To do**

**720p YUY**

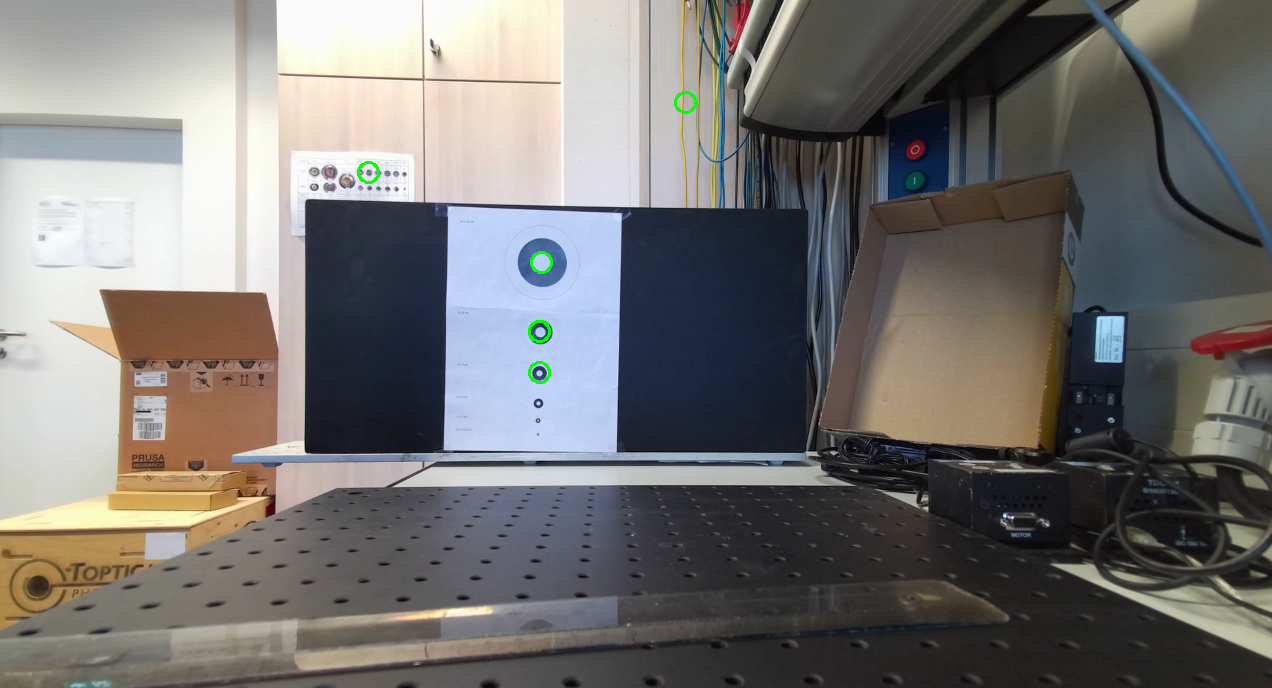
**400 mm**

****

**600mm**

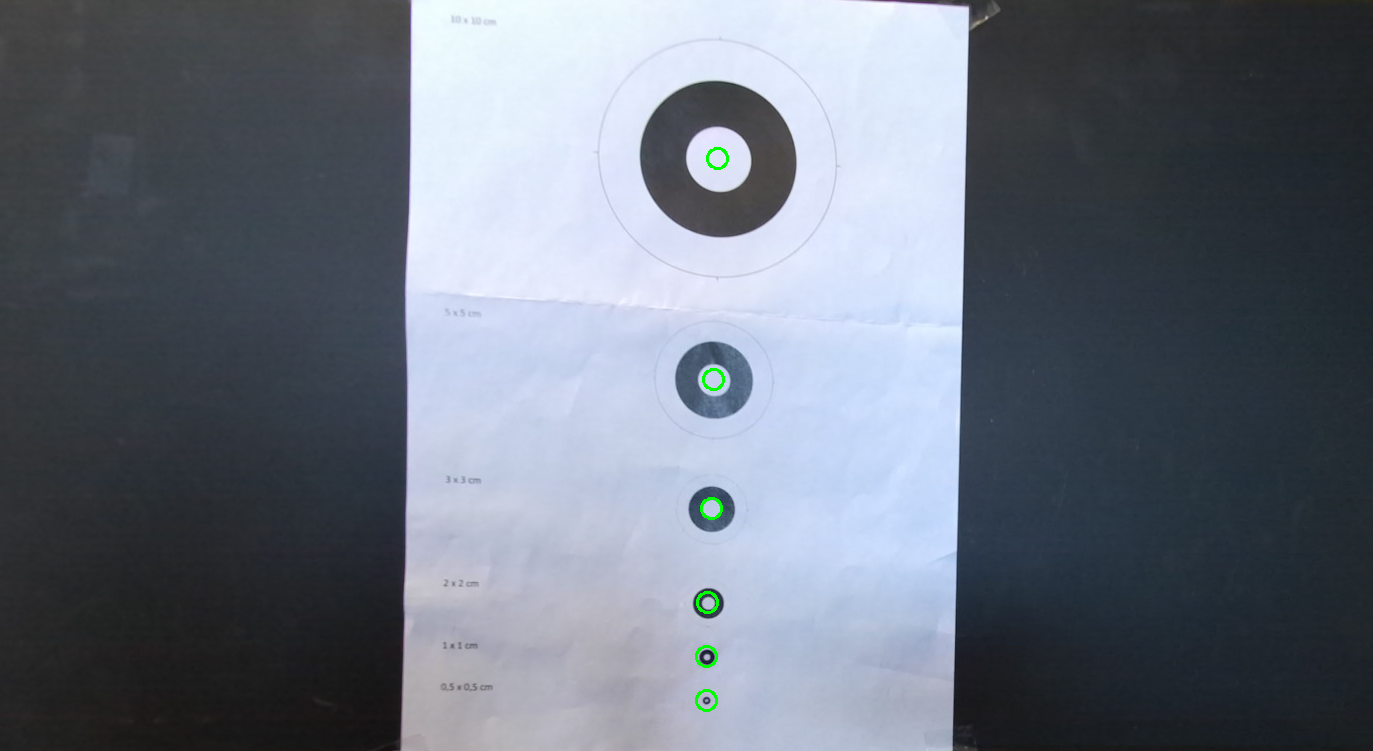
****

**800mm**

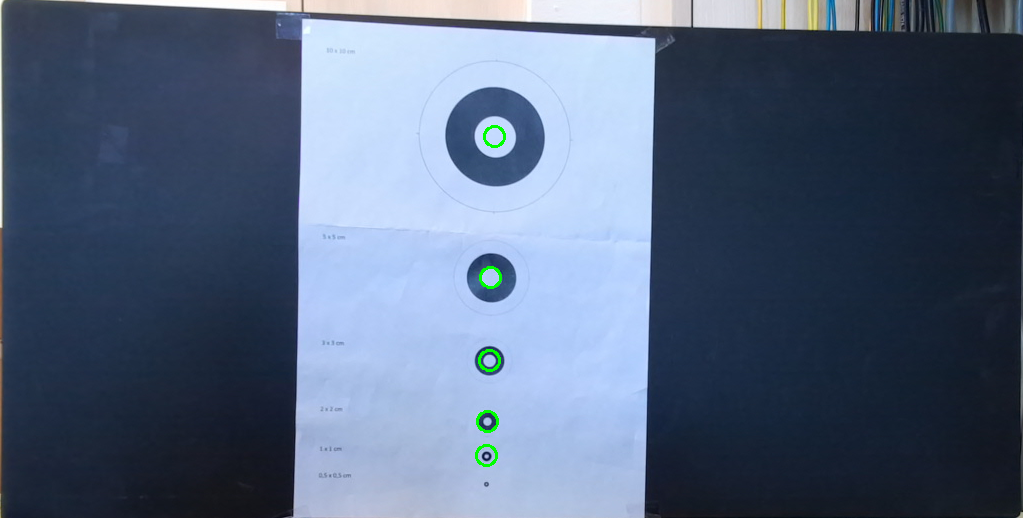
****

**1080p BGRA**

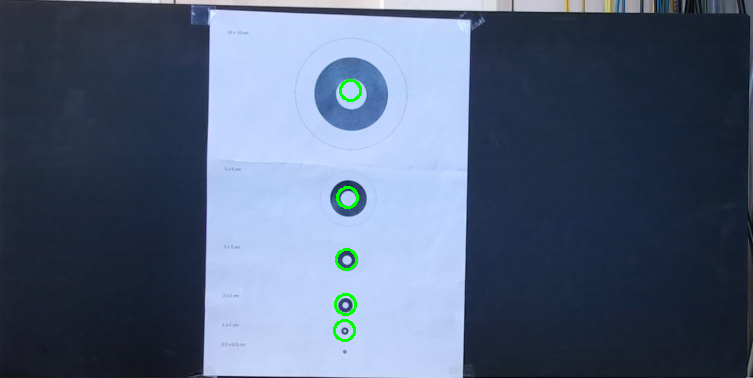
**400mm**

**qq**

**600mm**

****

**800mm**

****

**Tracking Performance**

-add photo of test

-describe test

-lag amount

-resulting difference in position

**Overall system schema**

**-Todo**

# References

|  |  |
| --- | --- |
| [1] | Optotune, "MR-E-2 Development Kit Operation Manual," Dietikon, 2019. |
| [2] | C. Simon, "simonensemble," [Online]. Available: https://simonensemble.github.io/posts/2018-10-27-orthogonal-procrustes/. |
| [3] | N. Ho, "FINDING OPTIMAL ROTATION AND TRANSLATION BETWEEN CORRESPONDING 3D POINTS," [Online]. Available: https://nghiaho.com/?page\_id=671. |
| [4] | "Kabsch algorithm," [Online]. Available: https://en.wikipedia.org/wiki/Kabsch\_algorithm. |
| [5] | "Azure Kinect Sensor SDK," Microsoft, [Online]. Available: https://microsoft.github.io/Azure-Kinect-Sensor-SDK/master/structk4a\_1\_1calibration\_a84577df64d47642d0b8f1fee11b21a96.html. [Accessed 08 08 2023]. |
| [6] | P. D. a. A. J. R. N. José Ferrão, "Detection of Aruco Markers Using the Quadrilateral Sum Conjuncture," Springer International Publishing. |