HoloLens Robot Controller

Düzceker Arda, Hein Jonas, Ktistakis Sophokles, Siller Quintanilla Rodolfo Octavio

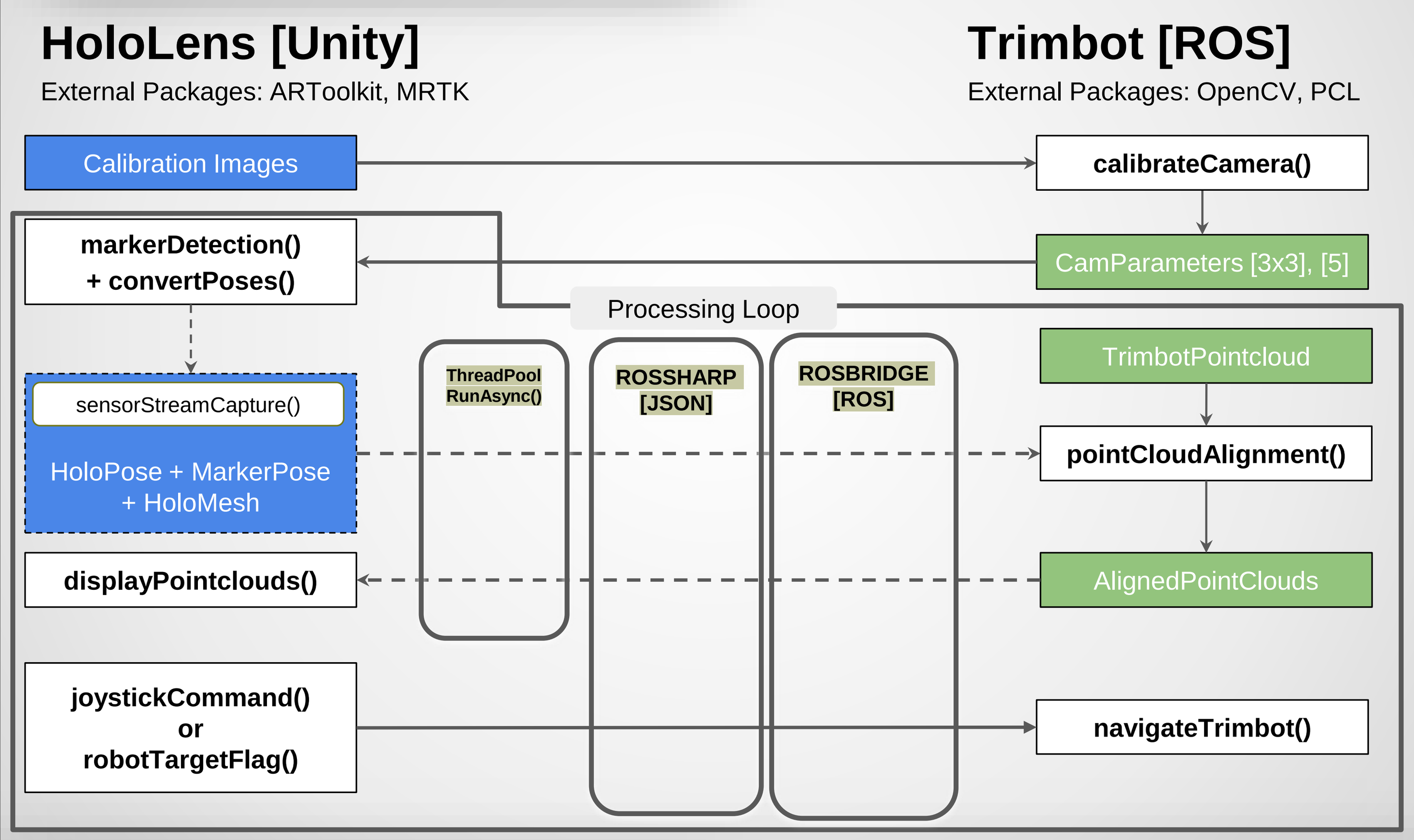
3D Vision, Spring 2019, ETH Zurich

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

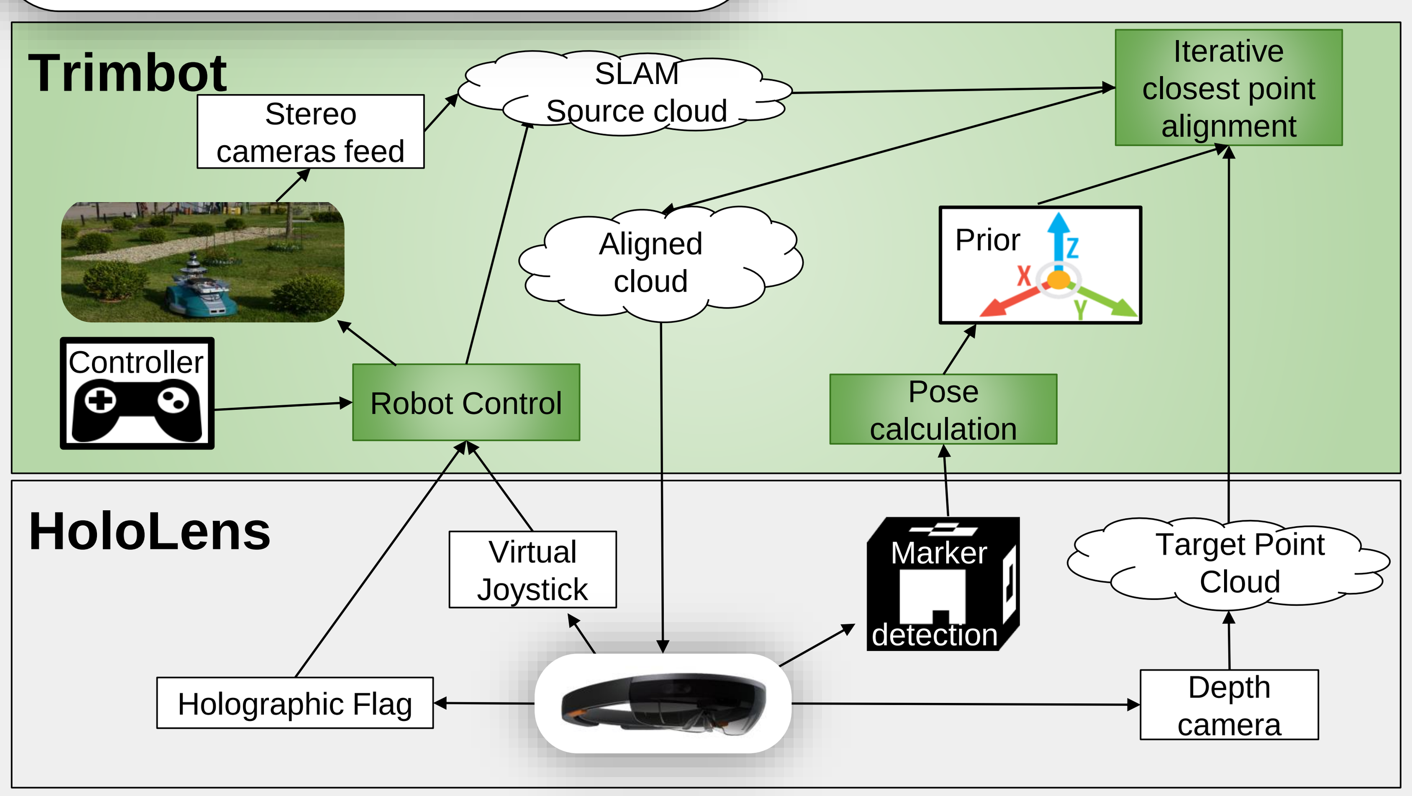
Introduction and Goals

- **An interaction system that displays the Trimbots generated SLAM point cloud as a hologram around the user in the HoloLens and enables him to control the Trimbot via holographic joystick commands or by pointing his fingers on a target location**
- The HoloLens will display the map acquired through robot's visual SLAM system as an overlay which enables user to perceive the state of the robot's visual system.
- For this, it is necessary to establish a communication interface between HoloLens and Trimbot which will be used to exchange map data stored in the respective devices as well as user's control commands.
- Also, the HoloLens needs to detect the robots pose and orientation in order to create the necessary coordinate transformations between the two systems.

Communication



Processing Pipeline



HoloLens App

I) Camera Calibration

- We follow the procedure by Zhang[5] and estimate the **camera calibration matrix** and **distortion coefficients** by taking multiple images of the calibration pattern under different orientations.

Homography between 3D points and the image plane

$$s\,m' = A[R|t]M'$$

$$s\begin{bmatrix}u\\v\\1\end{bmatrix}=\begin{bmatrix}f_x & 0 & c_x\\0 & f_y & c_y\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}r_{11} & r_{12} & r_{13} & t_1\\r_{21} & r_{22} & r_{23} & t_2\\r_{31} & r_{32} & r_{33} & t_3\end{bmatrix}\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix}$$

X, Y, Z: coordinates of a 3D point in world coordinate space [m]
u, v: image coordinates of the projected point [px]
A: camera matrix with intrinsic camera parameters
cx, cy: principal point at the image center [px]
fx, fy: focal lengths [px]

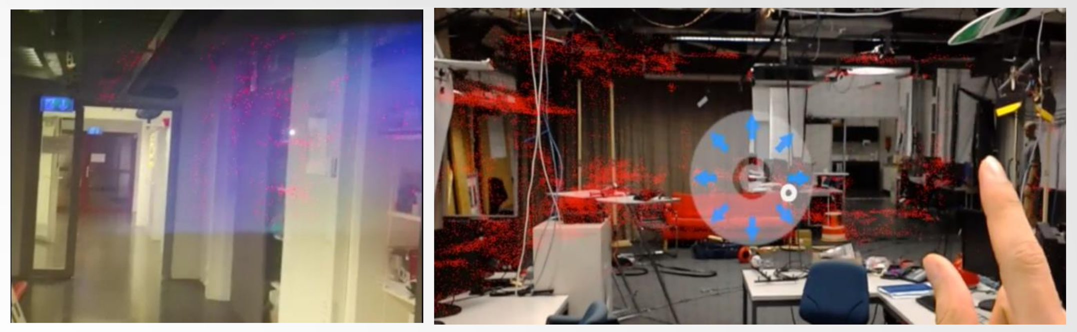
II) Marker Detection and Pose Estimation

Candidate Detection → Multi-Marker Identification → Pose Estimation

- Candidate Detection: Analyze the image (thresholding, contour extraction and selection) for square shapes that could be marker candidates
- Multi-Marker Identification: Apply perspective transformation and thresholding to extract black and white bits and identify markers
- Pose Estimation:
  - 3D transformation (rotation and translation) from marker to camera coordinate system.
  - Infer robot pose and orientation from identified markers

III) Point Cloud Rendering

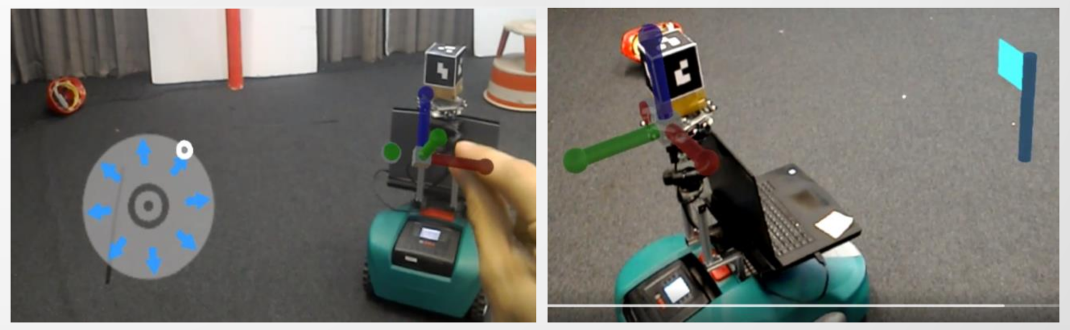
- **Renders** the most recently received **aligned point cloud mesh** published by the Trimbot
- If Trimbot **loses tracking** and **stops publishing** a new point cloud, user is instructed to **use the joystick** to help it **regain orientation**



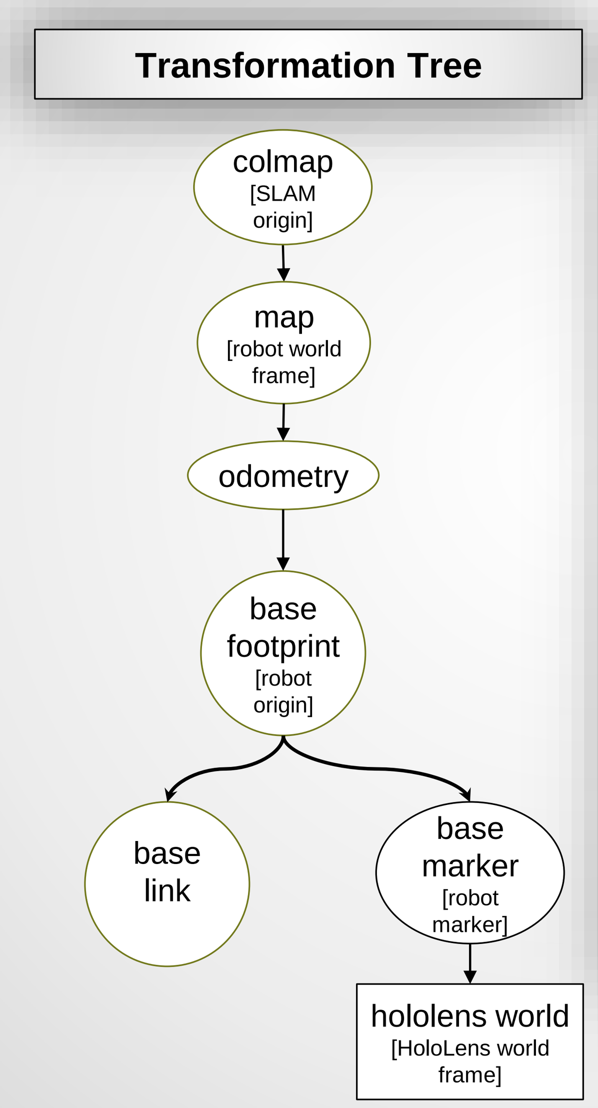
IV) Robot Control Commands

Option A:
- Send **direct control** commands to the Trimbot with a **virtual joystick**

Option B:
- Send **indirect control** commands by tapping at a **desired location** in the real world, which posts a **holographic flag** at that position.
- The **target location** is communicated to the Trimbot, which will **transform** it into its own coordinate system and **navigate autonomously towards the target.**



Transformations



The detected marker pose is used to **update the transformation** between the **robot's marker** and the **HoloLens world frame.**

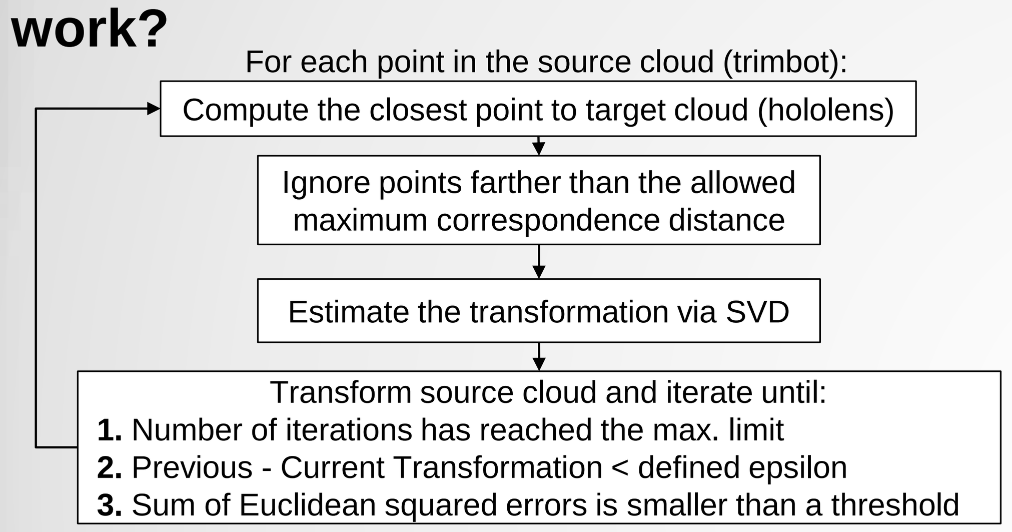
Then, the **overall transformation** is achieved by multiplying all the transformations in the tree from the SLAM origin down to the HoloLens.

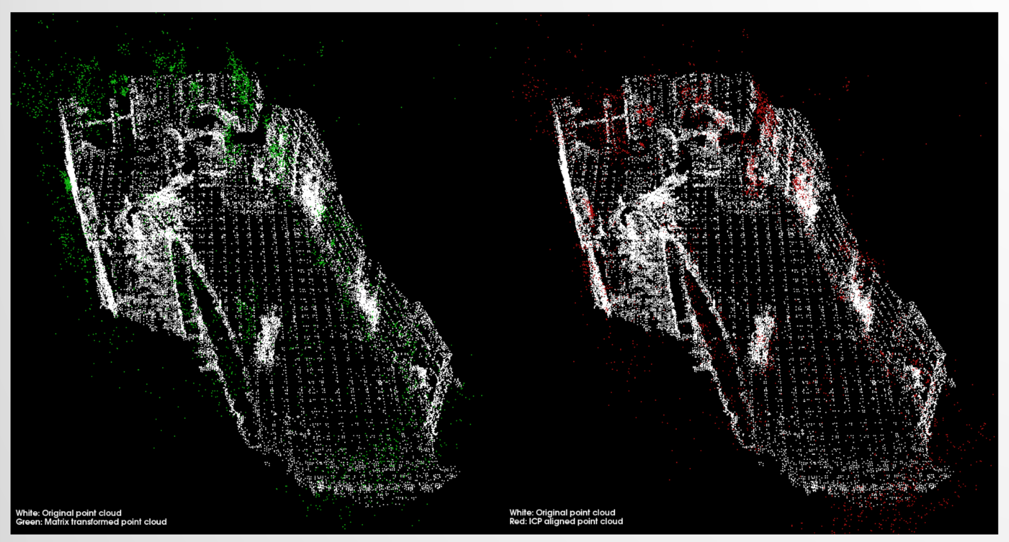
Problem:
If the trimbot **moves** but the marker pose **fails to update**, for example when the markers are not visible to the cameras of the HoloLens, then the transformation becomes invalid.

Solution:
We calculate the transformation only **after updating** the marker pose and **store it externally.**

Alignment

How does Iterative Closest Point (ICP) work?



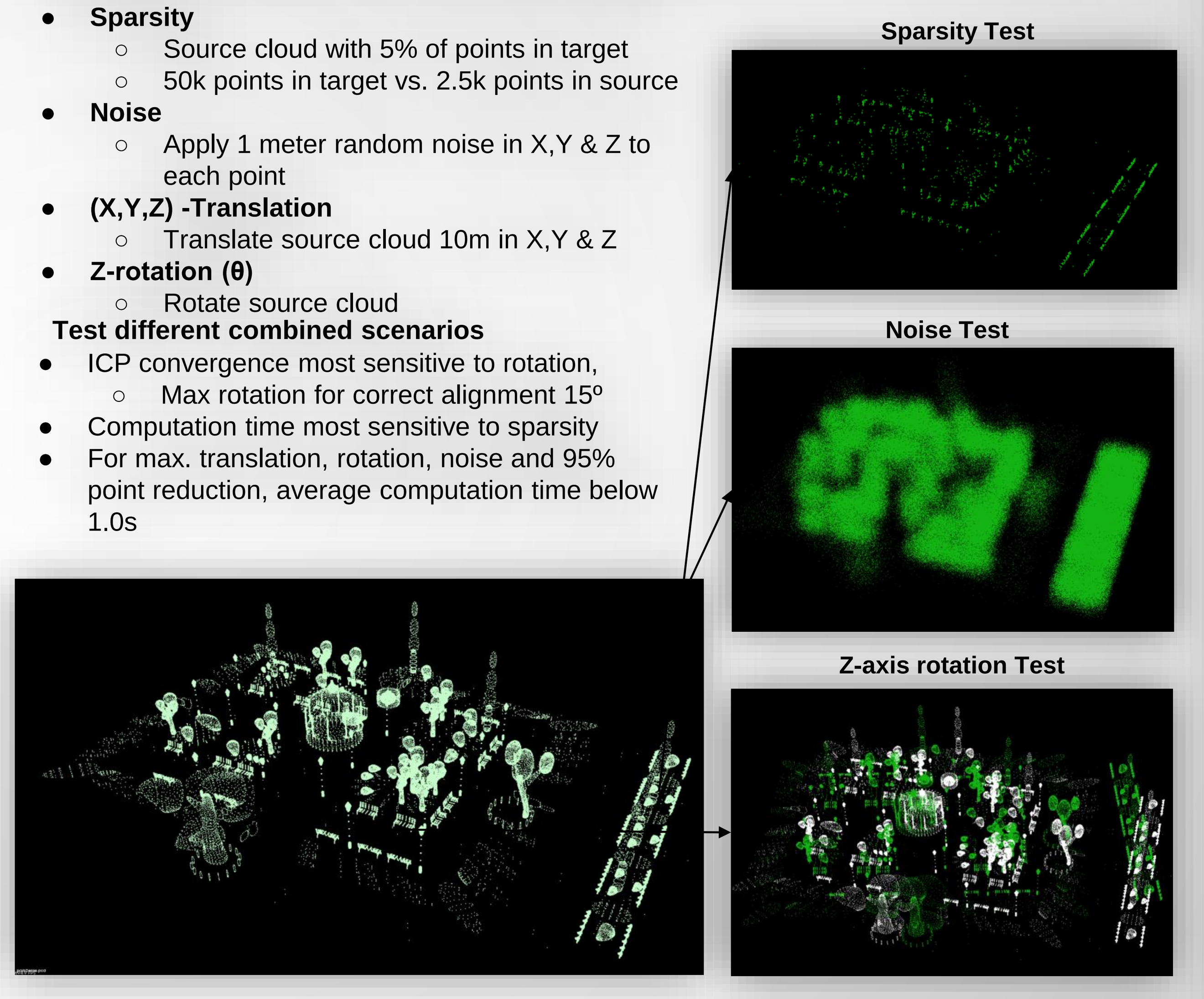


Synthetic data set tests

Without an available dataset to work with, a synthetic point cloud was used to test the limitations of the ICP algorithm. The target cloud was fixed and tested with different scenarios transforming an identical source cloud.

- **Sparsity**
  - Source cloud with 5% of points in target
  - 50k points in target vs. 2.5k points in source
- **Noise**
  - Apply 1 meter random noise in X,Y & Z to each point
- **(X,Y,Z) -Translation**
  - Translate source cloud 10m in X,Y & Z
- **Z-rotation (θ)**
  - Rotate source cloud

Test different combined scenarios
- ICP convergence most sensitive to rotation,
  - Max rotation for correct alignment 15º
- Computation time most sensitive to sparsity
- For max. translation, rotation, noise and 95% point reduction, average computation time below 1.0s



References

[1] J. Yang, H. Li, and Y. Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In ICCV, 2013
[2] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. TPAMI, 14(2):239–256, 1992
[3] Point Cloud Library (PCL): Pcl::IterativeClosestPoint< PointSource, PointTarget, Scalar > Class Template Reference, docs.pointclouds.org/trunk/classpcl_1_1_iterative_closest_point.html
[4] The Trimbot2020 project. http://trimbot2020.webhosting.rug.nl/
[5] Zhengyou Zhang. A Flexible New Technique for Camera Calibration, 1998
[6] Microsoft Mixed Reality Toolkit for Unity. https://microsoft.github.io/MixedRealityToolkit-Unity/Documentation/GettingStartedWithTheMRTK.html and https://github.com/microsoft/MixedRealityToolkit-Unity
[7] Microsoft HoloLens (1st gen) and immersive headset tutorials. https://docs.microsoft.com/en-us/windows/mixed-reality/holograms-100
[8] Long Qian. HoloLens ARToolKit:Marker tracking using the front-facing camera of HoloLens and Unity3D, with a wrapper of ARToolKit built for UWP. https://github.com/qian256/HoloLensARToolKit and http://longqian.me/2017/01/20/artoolkit-on-hololens/

Computer Vision and Geometry Lab