

Computer Vision: Part 2 (3D CV)

Dr. Nathanael L. Baisa

Lecture Content

- ▶ Introduction
- ▶ Camera Calibration
- ▶ 3D Reconstruction
- ▶ Point Clouds
- ▶ Point Clouds Data Acquisition
- ▶ Point Clouds for Robotics
- ▶ Conclusion
- ▶ References

Session Outcomes

- ▶ Gain a brief introduction to 3D computer vision.
- ▶ Acquire basic knowledge of camera calibration and 3D reconstruction.
- ▶ Understand what point cloud is and how it can be acquired.
- ▶ Get understanding of point clouds applications in robotics.

Introduction

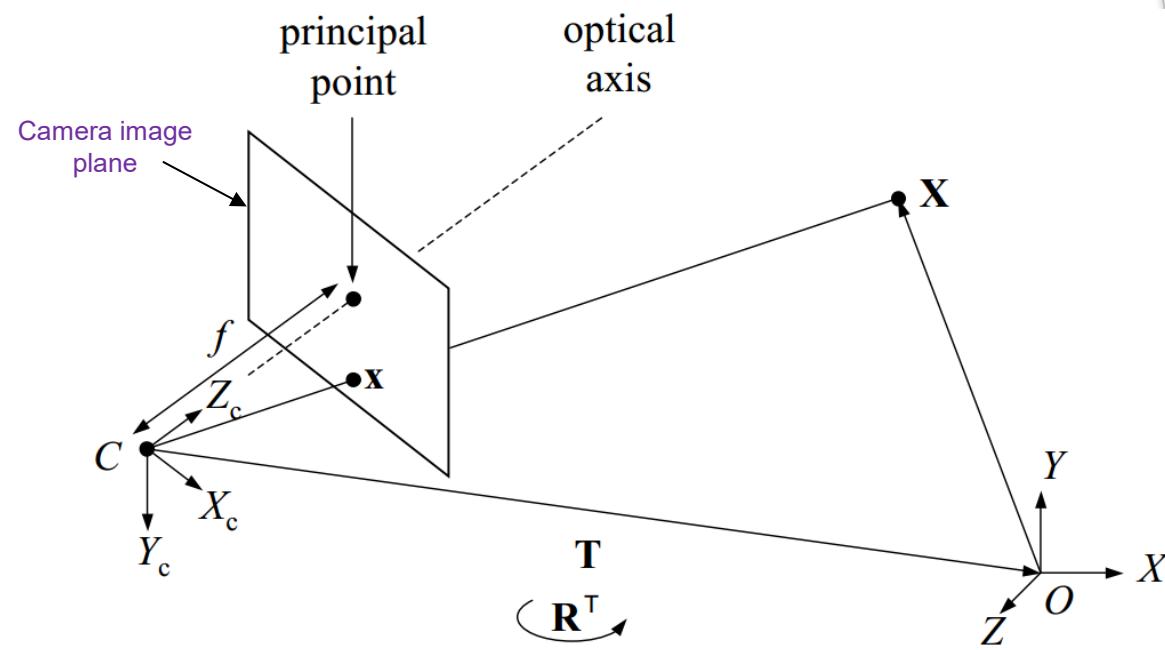
- ▶ We see **2D images** but perceive the **world** in **3D**.
- ▶ **3D reconstruction** capability is required in:
 - Intelligent robots.
 - Autonomous vehicles (self-driving cars).
 - Augmented and virtual reality.

Camera Calibration

- ▶ Camera calibration is the process of estimating the **camera parameters**:
 - **Intrinsic parameters:** The effective focal length, principal point (optical center) of the image plane, skew and lens distortion i.e. optics and internal geometry of the camera.
 - **Extrinsic parameters:** Camera pose (rotation and translation) with respect to the world coordinate system.

Camera Calibration

- ▶ $[u \ v \ 1]^T \sim \mathbf{P}[X \ Y \ Z \ 1]^T$
- ▶ $\mathbf{P} \sim \mathbf{K}[\mathbf{R} \ \mathbf{T}]_{3 \times 4}$
 - \mathbf{K} – Intrinsic matrix.
 - $[\mathbf{R} \ \mathbf{T}]$ – Extrinsic matrix.
 - \mathbf{P} – Projection Camera matrix.
- ▶ C – Camera coordinate system.
- ▶ O – World coordinate system.
- ▶ f is focal length.
- ▶ Pinhole projection of a 3D point \mathbf{X} onto a camera image plane.



Camera Calibration

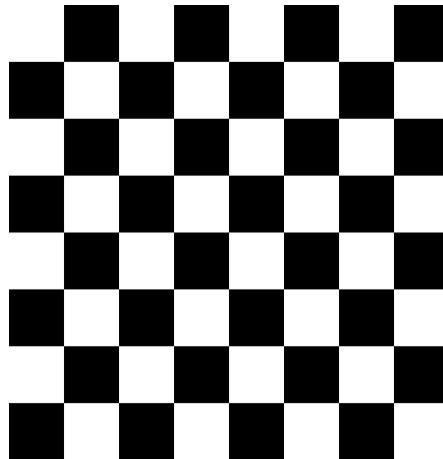
- ▶ \mathbf{K} is an upper triangular intrinsic camera calibration matrix of the form:

$$\mathbf{K} = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- ▶ where α_u and α_v are **scale factors**, s is **skew**, and $\mathbf{u}_0 = [u_0 \ v_0]^T$ is the **principal point**. Usually, **pixels** are assumed to be square in which case $\alpha_u = \alpha_v = \alpha$ and $s = 0$. Hence, α can be considered to be the **focal length** of the lens expressed in units of the **pixel dimension**.
- ▶ The **principal point** is where the **optical axis** intersects that **camera's image plane**.

Camera Calibration

- ▶ 3D world points (at least 6) and their corresponding 2D image points are needed e.g. using multiple images of a calibration pattern such as checkerboard.



- ▶ Once the projection matrix $\mathbf{P}_{3 \times 4}$ has been estimated, the first 3×3 sub-matrix can be decomposed (by QR decomposition i.e. $\mathbf{P}_s = \mathbf{K}\mathbf{R}$) into an upper triangular intrinsic camera calibration matrix \mathbf{K} and an orthonormal rotation matrix \mathbf{R} .

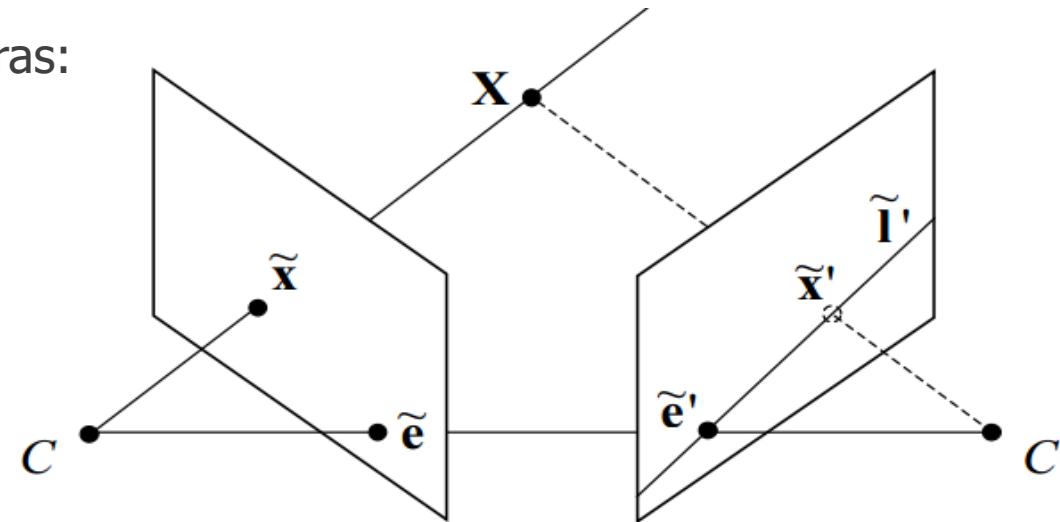
3D Reconstruction

- ▶ Camera systems calibration.
- ▶ Feature extraction.
- ▶ Feature points correspondence through matching.
- ▶ Reconstruction algorithms:
 - 2 views – Stereo vision.
 - N views – Sequential or batch methods.

Two-image 2D-to-3D Reconstruction

Method: Stereo Vision

- ▶ Epipolar geometry for two cameras:
 - Baseline – CC'
 - Epipoles - $\tilde{\mathbf{e}}$ and $\tilde{\mathbf{e}}'$
 - Epipolar plane - $\mathbf{C}\mathbf{X}\mathbf{C}'$
 - \mathbf{C} and \mathbf{C}' are camera centers.



- ▶ Two cameras taking the image of **the same scene**.
- ▶ **Epipole** is the point of intersection of line through **camera centers** and the **image planes**.
- ▶ Given the projection $\tilde{\mathbf{x}}$ of a 3D point \mathbf{X} in one image, its projection $\tilde{\mathbf{x}}'$ in a second image is **restricted** to the corresponding **epipolar line** \tilde{l}' .

Two-image 2D-to-3D Reconstruction

Method: Stereo Vision

- ▶ **Epipolar constraint:** Essential matrix ($\mathbf{E}_{3 \times 3}$) relates corresponding **image points** (at least 8) in two views:

$$\tilde{\mathbf{x}}^T \mathbf{E} \tilde{\mathbf{x}}' = 0, \quad \mathbf{E} \sim [\mathbf{T}]_{\mathbf{x}} \mathbf{R}$$

- ▶ $[\mathbf{T}]_{\mathbf{x}}$ is the cross-product matrix, where:

$$\text{For } \mathbf{T} = [t_x \quad t_y \quad t_z]^T, [\mathbf{T}]_{\mathbf{x}} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}.$$

- ▶ This epipolar constraint can be expressed using **pixel positions**:

$$\tilde{\mathbf{x}} \sim \mathbf{K}^{-1} \tilde{\mathbf{u}} \Rightarrow \tilde{\mathbf{u}}^T \mathbf{F} \tilde{\mathbf{u}}' = 0,$$

Two-image 2D-to-3D Reconstruction

Method: Stereo Vision

- ▶ $\mathbf{F}_{3 \times 3} \sim \mathbf{K}^{-1T} \mathbf{E} \mathbf{K}'^{-1}$ is the **fundamental matrix**.
- ▶ Fundamental Matrix **F** maps a **point** in one image to a line (**epiline**) in the other image. This is calculated from **matching points** from both images.
- ▶ NOTE:
 - ▶ The fundamental matrix **F** is just like the essential matrix **E**, except that **F** operates in image pixel coordinates whereas **E** operates in physical coordinates.
 - ▶ Calibrated cameras → Essential matrix.
 - ▶ Un-calibrated cameras → Fundamental matrix.

Two-image 2D-to-3D Reconstruction

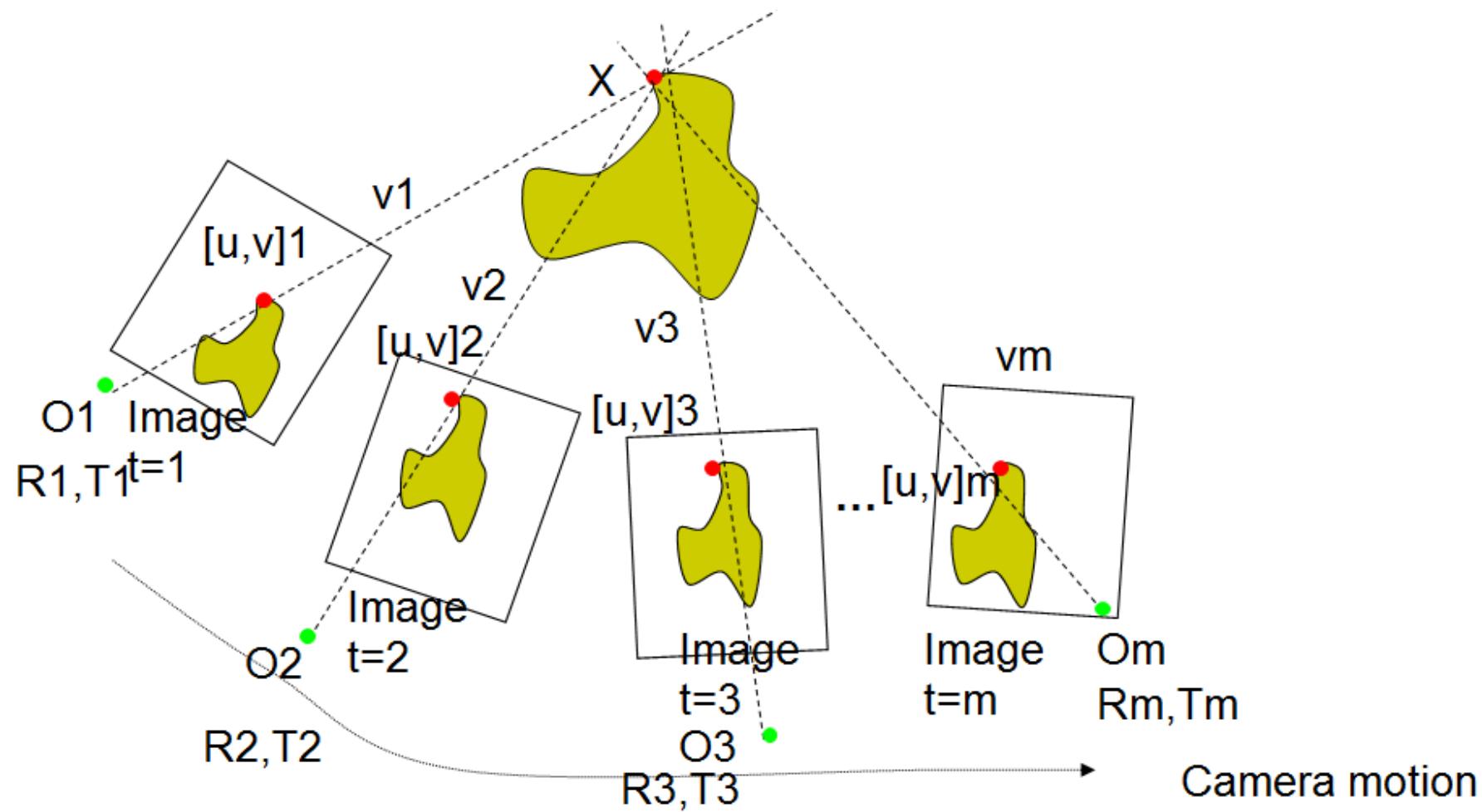
Method: Stereo Vision

- ▶ If the matrices \mathbf{K} and \mathbf{K}' are known, the recovered \mathbf{F} can be transformed into \mathbf{E} as

$$\mathbf{E} \sim \mathbf{K}'^T \mathbf{F} \mathbf{K}$$

- ▶ The essential matrix \mathbf{E} can be decomposed into \mathbf{R} and \mathbf{T} using singular value decomposition (SVD).
- ▶ The projection matrices (\mathbf{P} and \mathbf{P}') can be obtained from the recovered \mathbf{R} and \mathbf{T} and the known \mathbf{K} and \mathbf{K}' .
- ▶ Given projection matrices, 3D points can be computed from their measured image positions in two or more views → Triangulation.

N-image 2D-to-3D Reconstruction Method



N-image 2D-to-3D Reconstruction

Method: Sequential Method

- ▶ Order of images are used like in a **move**.
- ▶ Incorporating successive views one at a time.
- ▶ One possibility is to exploit the **two-view epipolar geometry** that relates each view to its predecessor.
- ▶ E.g. Known intrinsic parameters, **essential matrices** are estimated linearly using **8 or more points correspondences** (e.g. using **eight-point algorithm**) and decomposed to give **relative camera orientation** and the direction of camera **translation**. And then 3D points are estimated.

N-image 2D-to-3D Reconstruction

Method: Sequential Method

- ▶ From image features $\mathbf{u}_{ij} = (u_{ij}, v_{ij})$, structure from motion (SFM) gives an **initial estimate** of projection matrices \mathbf{P}_i and 3D points \mathbf{X}_j .
- ▶ Usually, it will be necessary to **refine this estimate** using iterative **non-linear optimisation** to minimize an appropriate **cost function** (a weighted sum of squared **re-projection errors** between the 2D image locations of detected (measured) and predicted image points, where the predicted points are re-projected from 3D structures by camera parameters) → **bundle adjustment**.

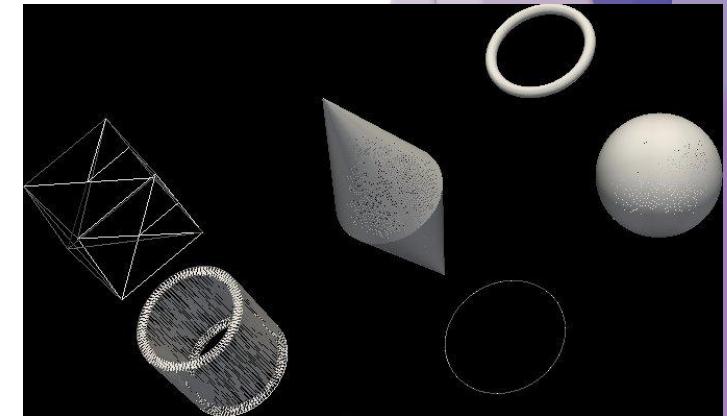
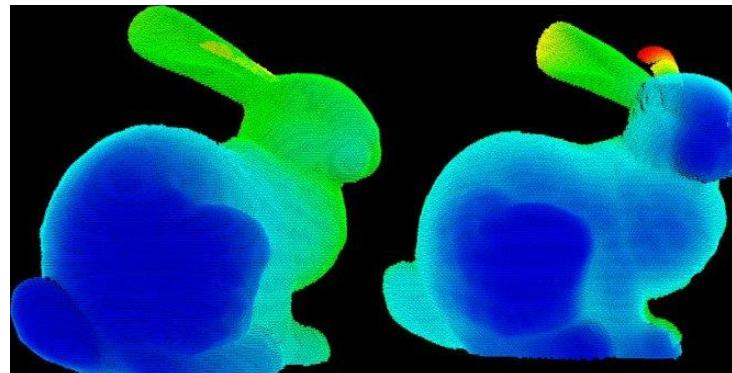
N-image 2D-to-3D Reconstruction

Method: Batch Method

- ▶ Order of images can be **random**.
- ▶ Bundle adjustment approach:
 - Guess iteratively the solution for **3D** to explain the measurements of **feature points** in all images.
 - A typical **non-linear optimization** problem.
 - **Gauss-Newton** for non-linear optimization method can be used.

Point Clouds

- ▶ A **point cloud** is a discrete set of data **points** in **space**. The points may represent a **3D shape** or **object**. Each point position has its set of Cartesian coordinates (X, Y, Z).
- ▶ XYZ, NxNyNz (normals) and RGB data could be available.



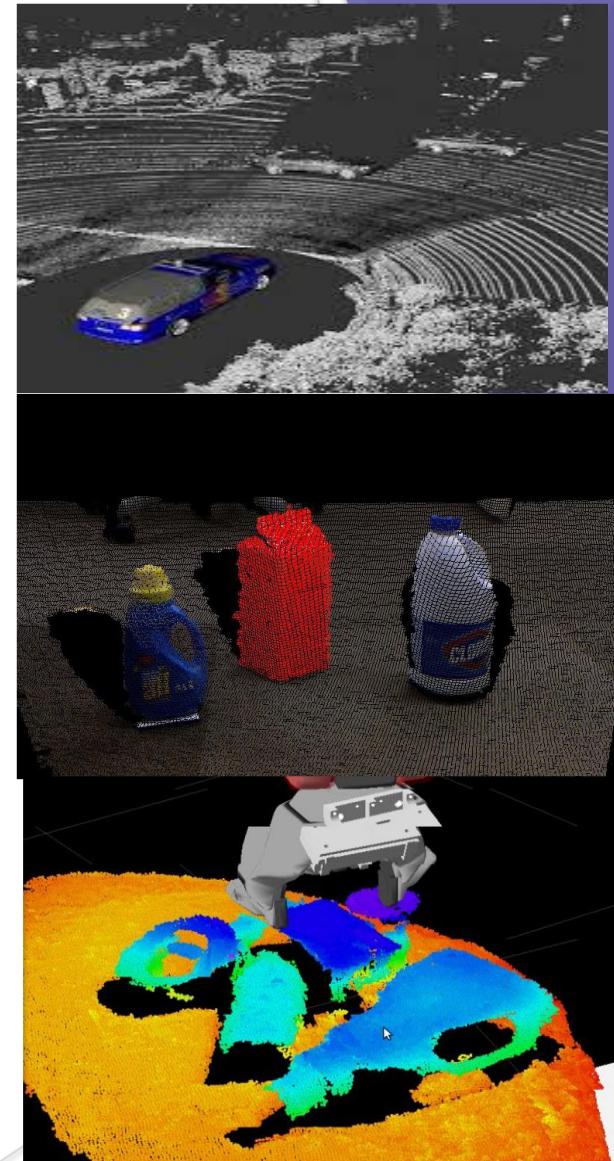
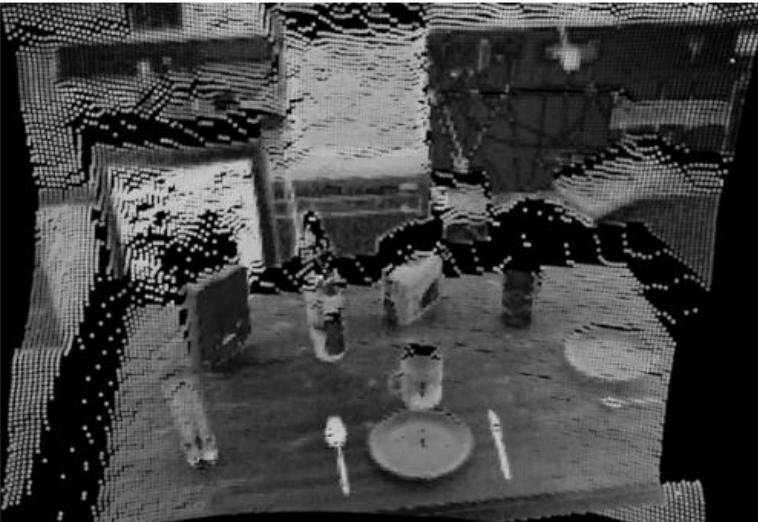
Point Clouds Data Acquisition

- ▶ Laser scans (high quality).
- ▶ Stereo cameras (passive & fast but dependent on texture).
- ▶ Time of flight cameras (fast but not as accurate/robust).
- ▶ Kinect-style Sensors such as Kinect, RealSense, etc.
- ▶ Computer generated.



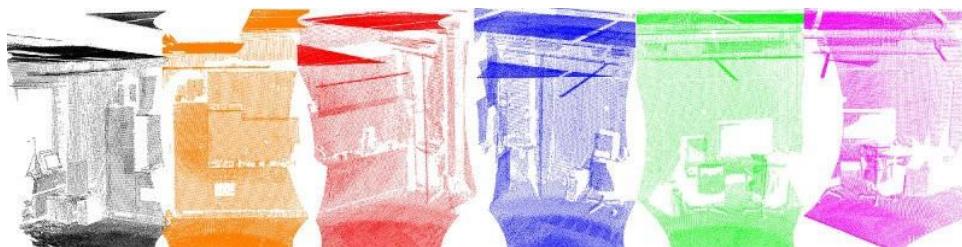
Point Clouds in Robotics

- ▶ Navigation / Obstacle avoidance.
- ▶ Object recognition, and registration for pose estimation using iterative closest point (ICP).
- ▶ Grasping and manipulation, etc.

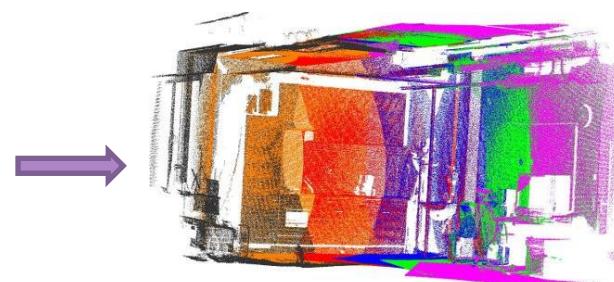


Point Clouds in Robotics

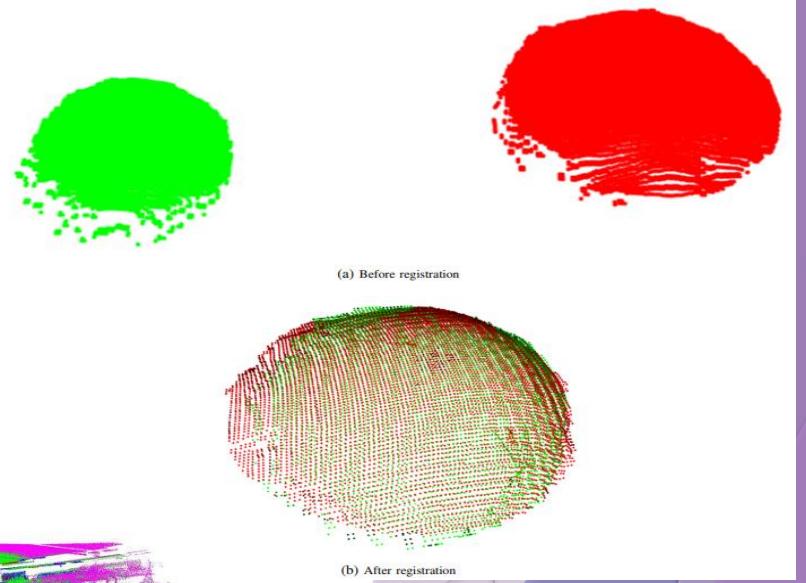
- ▶ Process **point clouds** by visiting <http://www.open3d.org/> such as
 - Finding number of points in a given point cloud raw data.
 - Visualization of point cloud data.
 - Downsampling point cloud data as well as removing outliers.
 - Extracting features from a given point cloud data, for instance, using **Fast Point Feature Histograms (FPFH)**.
 - Registering two point clouds data using **Iterative Closest Point (ICP)**.



Point clouds



Registered point cloud



Conclusion

- ▶ Though **3D scene** can be reconstructed from **2D camera images**, it is also possible to obtain **3D point clouds** directly using:
 - RGB-D sensors - combine RGB colour information with per-pixel depth information.
 - Kinect (structure light method – uses dots rather than strips).
 - Lidar (time of flight laser method).
 - Stereo camera using triangulation and bundle adjustment.
 - Monocular depth estimation using deep learning.
 - 3D reconstruction from monocular images using deep learning.
 - etc.
- ▶ 3D computer vision is very crucial for intelligent robotics applications.

References

- ▶ R. Szeliski, 'Computer Vision: Algorithms and Applications', Springer, 2021.
[\[https://szeliski.org/Book/\]](https://szeliski.org/Book/)
- ▶ R. Hartley and A. Zisserman, '*Multiple view geometry in computer vision*' Cambridge university press, 2003.
- ▶ <https://mi.eng.cam.ac.uk/~cipolla/publications/contributionToEditedBook/2008-SFM-chapters.pdf>
- ▶ https://docs.opencv.org/3.4/da/de9/tutorial_py_epipolar_geometry.html
- ▶ <https://www.cse.psu.edu/~rtc12/CSE486/lecture15.pdf>
- ▶ OpenCV: <https://opencv.org/>
- ▶ Open3D: <http://www.open3d.org/>
- ▶ <https://huggingface.co/learn/computer-vision-course/en/unit0/welcome/welcome>
- ▶ <https://github.com/polygon-software/python-visual-odometry>
- ▶ <https://github.com/luigifreda/pyslam>
- ▶ <https://github.com/alyssaq/3Dreconstruction>