Introducing a novel approach to analyse 6D Pose estimators under disturbances

Tobias Niedermaier

Master Thesis in Computer Science
Faculty of Electrical Engineering and Computer Science
University fo Applied Sciences Ravensburg-Weingarten

Matriculation number	32900
Supervisor	Prof. Dr. Stefan Elser
External Examiner	Prof. Dr. Markus Reischl

3rd April 2025

Tobias Niedermaier:
Introducing a novel approach to analyse 6D Pose estimators under disturbances
Master Thesis, University fo Applied Sciences Ravensburg-Weingarten, 2025.

Eigenständigkeitserklärung

Hiermit versichere ich, Tobias Niedermaier,

1. dass ich die vorliegende Arbeit selbstständig und ohne unzulässige Hilfe verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie die wörtlich und sinngemäß übernommenen Passagen aus anderen Werken kenntlich gemacht habe.

2. Außerdem erkläre ich, dass ich der Hochschule ein einfaches Nutzungsrecht zum Zwecke der Überprüfung mittels einer Plagiatssoftware in anonymisierter Form einräume.

Weingarten, 3. April 2025

Tobias Niedermaier

Abstract

Current state-of-the-art methods for evaluating 6 degrees of freedom (6D) pose estimators have several significant limitations. Existing error metrics often yield near-zero errors even for inaccurate pose estimations and are highly dependent on the object point cloud used, leading to inconsistent results across different objects. Moreover, these metrics fail to account for false detections. Accurate evaluation of pose estimators is crucial for applications in robotics, augmented reality, and object manipulation, where reliable performance is essential. Evaluation is especially critical when analysing 6D pose estimators under disturbance, to gain insight on how the disturbances affect the pose estimator. This thesis introduces a novel error metric and evaluation score that can assess poses independently of the specific object and incorporate false detections. The proposed score is adjustable for various evaluation scenarios. A theoretical discussion, along with a use case analysing a 6D pose estimator under disturbances, demonstrates the advantages of the new evaluation method compared to existing state-of-the-art approaches.

Zusammenfassung

Die aktuellen Methoden zur Bewertung von 6 Freiheitsgrade (6D) Positionsschätzern weisen mehrere erhebliche Einschränkungen auf. Bestehende Fehlermetriken liefern oft Fehler nahe Null, selbst bei ungenauen Posenschätzungen, und sind stark von der verwendeten Objektpunktwolke abhängig, was zu uneinheitlichen Ergebnissen bei verschiedenen Objekten führt. Zudem berücksichtigen diese Metriken keine falschen Erkennungen. Eine präzise Bewertung von Posenschätzern ist jedoch entscheidend für Anwendungen in der Robotik, erweiterten Realität und Objektmanipulation, bei denen zuverlässige Leistung unerlässlich

ist. Dies gilt besonders, wenn 6D-Positionsschätzer unter Störungen analysiert werden, um zu verstehen, wie sich diese Störungen auf den Posenschätzer auswirken. In dieser Arbeit werden eine neuartige Fehlermetrik und eine Bewertungsskala vorgestellt, die Posen unabhängig vom spezifischen Objekt bewerten und falsche Erkennungen einbeziehen können. Die vorgeschlagene Bewertungsmethode ist anpassbar und kann somit auf verschiedene Bewertungsszenarien abgestimmt werden. Eine theoretische Diskussion sowie ein Anwendungsfall, der einen 6D-Positionsschätzer unter Störungen analysiert, verdeutlichen die Vorteile der neuen Bewertungsmethode im Vergleich zu bestehenden Ansätzten nach dem Stand der Technik.

Acknowledgements

I would like to thank Christopher Bonenberger, Stephan Scholz, Felix Berens and Tobias Spohn for their invaluable support throughout the course of this thesis. Their meticulous proofreading and insightful discussions have significantly contributed to the refinement of this work.

Contents

List of Figures

List of Tables

List of Abbreviations

6D 6 degrees of freedom; Three from the 3D position and another three from the rotation in 3D space. 3, 4, 11, 12, 21, 26, 36, 39, 59

ACPD Average Corresponding Point Distance; A metric for 6D pose estimation, using corresponding point distances of the estimated and every possible ground truth point cloud. It is defined in Section 3.1.6. xiv, xv, 15, 21–24, 26–32, 35, 37

ADD Average Distance of model points; A metric for 6D pose estimation, using corresponding point distances of the estimated and ground truth point cloud. xiii–xix, 4, 7, 13, 15, 17, 20–22, 24, 26–33, 35, 37, 39–45, 47, 48, 50–57, 59

ADDS Average Distance of model points for Symmertic Objects; A metric for 6D pose estimation, using minimal point distances of the estimated and ground truth point cloud. It is defined in Section 3.1.2. xiii–xix, 6–8, 14, 17, 20–24, 26, 28–30, 32–37, 39–45, 47, 48, 50–54, 56, 57

AIMRTES Average Inverse Multi Rotation Translation Error Score; A score for 6D pose estimation, using the average inverse of the multi rotation translation error. It is defined in Section 3.2.5. xv–xix, 34, 37, 39–45, 47, 48, 50–57, 59

AUC Area under curve; A score for 6D pose estimation, using the integral of the accuracy, over varying thresholds. It is defined in Section 3.1.11. xiii, xv–xix, 7, 17, 20–22, 31, 33, 34, 36, 37, 39–42, 44, 47, 48, 50–57

CoU Complement over Union; A metric for 6D pose estimation, which is defined as the complement of the unified area/volume of the estimated and ground truth object. It is defined in Section 3.1.5. 18, 34, 35

FFB6D Full Flow Biderectional Fusion Network for 6D Pose Estimation; A 6D pose estimator which at its core uses a neural network to find poses. It uses full flow fiderectional fusion between its convelutinoal layers to fuse

RGB and depth data. It is described in Section 2.1.5. xiii, xvi–xix, 5, 8, 17, 39, 41–45, 48, 50, 55–57, 59, 60

ICP Iterative Closed Point; An aglorithem which fits two point clouds onto each other, recovering the rotation and translation needed to move one point cloud into the other. It is defined in Section 2.1.4. 5, 7, 8

MCPD Maximum Corresponding Point Distance; A metric for 6D pose estimation, using maximum point distances of the estimated and every possible ground truth point cloud. It is defined in Section 3.1.7. xiv, xv, 21–24, 26–32, 35, 37

MR Mean recall; A score for 6D pose estimation, using the average accuracy over multiple thresholds. It is defined in Section 3.1.10. 17, 36

MRE Multi rotation error; A metric for 6D pose estimation, which only measure the rotation of the estimated and ground truth object, while considering symmetries and rotation invariance of an object. It is defined in Section 3.2.3. xvi, xvii, xix, 20, 40–43, 45, 47, 48, 50

MRTE Multi rotation translation error; A metric for 6D pose estimation, which combines and scales the multi rotation and translation error. It is defined in Section 3.2.4. xiv, 22–25, 31, 37, 40, 59

PCA Principal component analysis; A statistical method used to identify symmetry axes in 3D point clouds by determining the primary directions of variance in the data.. 8, 9

VSD Visible Surface Discrepancy; A metric for 6D pose estimation, using the distance between the visable surface of the estimated and ground truth object. It is defined in Section 3.1.8. 16, 21, 35, 36

w. f.d. Without false detections. xvi, xviii, 42, 44, 45, 53–56

List of Symbols

$\|\cdot\|_F$ The Frobenius norm of a matrix. 6, 14

$\tilde{\mathbf{R}}_{\mathbf{u},\theta}$ A rotation matrix of size 3×3, which contains the rotation around axis $\mathbf{u}$ with angle θ. 18

A The ground truth 2D bounding box or segmentation mask of on object. 15

B The ground truth 3D bounding box or segmentation mask of on object. 15, 18

β The usability threshold used by the scaling function $g(x)$, after which the translation error is cut off. 20, 40

$\hat{A}$ The estimated 2D bounding box or segmentation mask of on object. 15

$\hat{B}$ The estimated 3D bounding box or segmentation mask of on object. 15, 18

$\hat{\mathbf{R}}$ A 3×3 rotation matrix, which contains the estimated rotation of an object. 4–7, 12–15, 19, 20, 22

$\hat{\mathbf{t}}$ A vector of size 3 containg the estimated translation of an object. 4–7, 12–15, 33

$\mathbf{0}$ A vector filled with zeros. xxiv, 6, 9

$\mathbf{1}$ A vector filled with ones. 6

$\mathbf{A}$ A matrix containing the points of a point cloud in its rows, every column represents one point. It is of size $3 \times |\mathcal{P}|$. 5–9

$\mathbf{B}$ A matrix containing the points of the ICP target point cloud in its rows, every column represents one point. For the purposes used in this thesis it is of size $3 \times |\mathcal{P}|$. 5–7

$\mathbf{C}$ The covariance matrix, used in Principal Component Analysis (PCA) to capture the pairwise covariances between the different features of a dataset. It is a symmetric matrix where each element $\mathbf{C}_{ij}$ represents the covariance

between the i-th and j-th features. The eigenvectors of the covariance matrix indicate the directions of maximum variance, which are used to define the principal components.. 9

$\mathbf{I}$ The identity matrix, a square matrix, which contains ones on its main diagonal and zero everywhere else. 7, 14, 19, 20

$\mathbf{K}$ The intrsic camera matrix. 9, 10

$\mathbf{R}$ A 3×3 rotation matrix, which contains the ground truth rotation of an object. xxv, 6, 7, 12–15, 19, 20

$\mathbf{T}$ A matrix, which contains a translation in its rows, i.e. every column is the same translation. This matrix is of size $3 \times |\mathcal{P}|$. 6, 7

$\mathbf{Y}$ A matrix containing the points of the ICP target point cloud in its rows, every column represents one point. It is of size $3 \times |\mathcal{P}|$ and its centre is at $\mathbf{0} \in \mathbb{R}^3$. 6, 7

$\mathbf{Z}$ A matrix containing the points of a point cloud in its rows, every column represents one point. It is of size $3 \times |\mathcal{P}|$ and its centre is at $\mathbf{0} \in \mathbb{R}^3$. 6, 7, 9

$\mathbf{p}$ A singe point in 3D space, represented as a vector of size 3. 3, 4, 12, 13, 15

$\mathbf{t}$ A vector of size 3 containg the ground truth translation of an object. xxv, 6, 7, 12–15

$\mathbf{u}$ A symmertic axis of an object. xxiv, 18, 19

$\mathbf{v}$ A symmetric axis $\mathbf{u}$ with previous rotations applied. 19

$\mathbf{w}$ A rotation invariant axis of an object. 19, 20

$\mathcal{A}$ The set which contains all symmetric axis of an object, with the previous rotations applied. 19

$\mathcal{B}_i$ The set which contains all correct rotation angles for the symmetric axis $\mathbf{u}_i$. 18, 19

$\mathcal{O}$ The set of all objects in a dataset. 3

$\mathcal{P}_m$ The set of all points in a point cloud, for an object with a multiple correct poses. 13, 15

$\mathcal{P}_s$ The set of all points in a point cloud, for an object with a single correct pose. 12

$\mathcal{P}_{\text{est}}$ The set of all points in a point cloud, with the estimated rotation and translation applied. 4, 12

$\mathcal{P}$ The set containg all points of a pointcloud. xxiii–xxv, 3, 4, 6, 9

$\mathcal{Q}$ The set containg all correct poses ($\mathbf{R}$, $\mathbf{t}$) of an object. 15, 16

$\mathcal{R}$ The set which contains all correct symmetric rotations, after the ground truth rotation is applied, i.e. all correct symmetric rotations of an object. 19, 20

$\mathcal{T}$ The set containg all thresholds τ used for calcuting the mean recall. 16, 36

$\overline{\mathbf{C}}$ A matrix, which contains the center vector $\overline{\mathbf{c}}$ in its rows, i.e. every column is the same center vector of a point cloud. This matrix is of size $3 \times |\mathcal{P}|$. 6, 9

$\overline{\mathbf{c}}$ A vector containing the center point of a point cloud. It follow that this vector is of size 3. xxv, 6

τ Error metric threshold after which an pose is no longer considered correct. xxv, 16, 17, 36

c_x The x-coordinate of the optical center (principal point) in the image plane, representing the horizontal offset from the origin of the image coordinate system.. 9

c_y The y-coordinate of the optical center (principal point) in the image plane, representing the vertical offset from the origin of the image coordinate system.. 9

e_{ADDS} The Average Distance of Model Points for Symmetric Objectes (ADDS) error metric, for a single object. Defined in Section 3.1.2. 13

e_{ADD} The Average Distance of Model Points (ADD) error metric, for a single object. Defined in Section 3.1.1. 12, 33

e_{MRTE} The multi rotation translation error (MRE) defined in Section 3.2.4. 20

e_{R} The multi rotation error (MRE) defined in Section 3.2.3. 20, 21

e_{r} The rotaion error (RE) metric, for a single object. Defined in Section 3.1.4. 14, 19, 24

e_{t} The translation error (TE) metric, for a single object. Defined in Section 3.1.3. 14, 20, 21

$f(x)$ The scaling function for the multi rotation error in the MRTE. In this thesis the default scaling is $f(x) = \frac{x}{2\sqrt{2}}$. 20

f_x The focal length of the camera in the x direction, measured in pixels. It represents the scaling factor that maps the horizontal dimension of 3D points in the camera coordinate system to the 2D image plane.. 9

f_y The focal length of the camera in the y direction, measured in pixels. It represents the scaling factor that maps the vertical dimension of 3D points in the camera coordinate system to the 2D image plane.. 9

$g(x)$ The scaling function for the translation error in the MRTE. In this thesis the default scaling is $g(x) = \min(\frac{x}{\beta}, \beta)$. xxiii, 20, 40

o A single object. 3

p_x Represents the x-coordinate of a pixel in the image, indicating its horizontal position within the image grid. 9, 10

p_y Represents the y-coordinate of a pixel in the image, indicating its vertical position within the image grid. 9, 10

1 Introduction

The task of detecting and finding poses of objects in three-dimensional (3D) space has gained significant attention in the recent decade. Successfully solving this task would represent a substantial leap forward in various applications, including autonomous driving, industrial and service robotics, and virtual reality, to name a few.

How these poses estimators are evaluated plays a crucial role in developing better pose estimators. Given the challenging nature of not only detecting the object, but also finding its pose, i.e. its rotation and translation. It is not immediately obvious how the evaluation process should work, since various use cases focus on different error aspects.

The main problem this thesis is trying to address is that no evaluation score is available which takes rotation, translation and detection rates into account. Additional currently deployed metrics are treating objects vastly different, depending on their point cloud, some objects can produce errors up to three times higher than others, under the same error in the pose estimation.

Since the quantitive results are often treated as of higher importance, than qualitative results, the calculation of the quantitive results is of upmost importance. In practise, pose estimation methods are developed with the goal to achieve a better quantitive score. If this score calculation does not take an error source into account, false detections for example, this aspect will most likely be ignored entirely, making the pose estimator unsuited for real applications.

For the use case of examining how a pose estimator performances under disturbances, the evaluation also plays a central role, because every error aspect should be considered. For pose estimators, a multisensory setup is often employed, which brings the need for sensor fusion. Depending on the fusion algorithm, an error in one sensor can propagate to other sensors, thereby compounding the overall error in pose estimation. For instance, if a sensor fusion method heavily relies on a particular sensor's data, an error in that sensor can disproportionately affect the combined estimate, leading to worse results than if the sensors were used independently. For real applications, knowing how a pose estimator performance under sensor disturbances is important, because disturbances can originate from various sources, including environmental conditions, sensor lim-

itations, and dynamic changes in the operational context. The current evaluation methods are not suited for examining performance under disturbances, since they do not take every kind of error into account and treat the pose estimation as a binary problem where a pose is either considered correct or incorrect. Only when a pose is considered correct, the quality of the pose is considered. Also, recovering the kind of error from the current metrics and scores is not possible. The question this thesis tries to answer is: How can pose estimators be evaluated, while considering every kind of error and being unbiased to different kind of objects? Also, how can pose estimators be analysed under disturbances, while being able to see how a disturbance affects different kinds of errors? As answering these questions makes it possible to develop pose estimators more suited for real applications.

1.1 THESIS STRUCTURE

In Chapter 2 preliminary information is given. In Chapter 3 current evaluation methods are introduced, alongside a new novel approach for evaluation. Additionally, a theoretical compression is given between the evaluation methods is given. In Chapter 4 the new novel approach, as well as the state-of-the-art evaluation is used on the use case of analysing a pose estimator under disturbances. In Chapter 5 a conclusion is drawn and questions for future work are raised.

2 Fundamentals

This chapter deals with preliminary information needed to understand this thesis.

2.1 6D POSE ESTIMATION

In this section, the task of 6 degrees of freedom (6D) pose estimation is explained in detail. The goal of a 6D pose estimation method is to find both the position (translation) and orientation (rotation) of predefined objects in three-dimensional space. Here, predefined objects refer to those for which a 3D scan, in the form of a high density point cloud and 3D textured model, is available. The point cloud of an object $o \in \mathcal{O}$ is denoted as the set $\mathcal{P}$ and contains every point $\mathbf{p}$ on the surface of a scanned object. If follows that the set $\mathcal{O}$ contains all objects in a given dataset.

2.1.1 COMMONLY USED INPUT

An RGB (red, green, blue) image is almost always used for pose estimation. This is due to the fact, that detecting objects on 2D images has been proven to work well. An example of state of the art 2D object detectors would be YOLO which is introduced in [24]. Also, RGB cameras are inexpensive and widely available, meaning RBG image data is included in most datasets, like the YCB-Video and LineMOD dataset, which are introduced in [36] and [13] respectively and a camera can easily be added to most sensor setups.
Reliably extracting the 3D position of objects, relative to a fixed point, is much harder. Especially when no markers are available. This is why a depth sensor, providing a point cloud or depth image, is often used alongside an RGB image. This greatly improves the accuracy, as can be seen when comparing state of the art 6D pose estimators using only RGB images with estimators using RGBD (red, green, blue, depth) data. And example of an RGBD image can seen in Figure 2.1. On the YCB-Video benchmark PoseCNN, which is introduced in [36], achieves

an increase of 25.6% mean ADD[1], from 53.7% to 79.3%.
While depth sensors are more expensive than RGB cameras, they are still commonly available. This makes RGBD data common in datasets, benchmarks and prototyping, which is why this thesis will focus on these data channels.
In theory, all information for a good pose estimation is given by RGBD data. When looking at PoseCNN the RGB image is used to detect object 2D bounding boxes, which are cropped out of the depth image, from which a good estimation of the object position can be obtained. The orientation is then found by fitting the object point cloud inside the cropped depth image.

2.1.2 OUTPUT

The pose estimator takes the input (like an RGBD image) and outputs the position and rotation for every detected object. The estimated rotation is denoted as $\hat{\mathbf{R}}$ and the translation $\hat{\mathbf{t}}$, for a single object. The rotation matrix $\hat{\mathbf{R}}$ is a 3×3 matrix and the vector $\hat{\mathbf{t}}$ is of size 3.
Pose estimators can also output poses for objects not in the input, this is known as a false detection. They can also not output a pose for an object in the input this is known as a missed detection or a false negative.
The output, meaning the estimated poses, can be visualized by applying the estimated rotation $\hat{\mathbf{R}}$ and translation $\hat{\mathbf{t}}$ to the point cloud of the object $\mathcal{P}$, resulting in the estimated point cloud $\mathcal{P}_{\text{est}}$,

$$\mathcal{P}_{\text{est}} = \left\{ \hat{\mathbf{R}}\mathbf{p} + \hat{\mathbf{t}} | \mathbf{p} \in \mathcal{P} \right\}. \tag{2.1}$$

2.1.3 YCB-VIDEO DATASET

The YCB-Video dataset is introduced in [36] and is one of the largest and most commonly used dataset for 6D pose estimation. It consists of 21 common household items, for each object a high density point cloud, as well as 3D texture scans are available. It provides RBG and depth images and ground truth poses for 133,827 frames, in 92 different scenes. An example frame can be found in Figure 2.1.

ASUS XTION

The Asus Xtion was used to record the YCB-Video dataset, the technical specifications are provided in [1]. For the YCB-Video dataset the 30 fps mode was used, which outputs 640×480 RGB and depth images. For the creation of the depth images, an infrared structured light sensor is used. How structured light

[1]The metrics will be explained in detail in the Chapter 3.

(a) RGB image.

(b) Depth image

Figure 2.1: An example of a frame from the YCB-Video dataset, both the RGB and the corresponding depth image are provided.

works in detail is explained in [26].
To summarize how structured light works: It projects a pattern onto an object's surface. The projected pattern deforms according to the surface geometry of the object and is reflected back to the sensor. From the deformed pattern a 3D model of the scanned surface is calculated.
Structured light systems, while effective for capturing detailed 3D information, can be susceptible to various disturbances that impact their accuracy and performance. One common disturbance is ambient light interference, where external light sources, especially those with infrared components like sunlight or certain artificial lights, can disrupt the projected pattern and lead to erroneous depth readings. Surface texture and colour can also affect pattern visibility. Highly textured or very dark surfaces may absorb the light, reducing contrast and complicating the detection of pattern deformations. Furthermore, motion blur can occur if the object or sensor is moving too quickly during capture, leading to distorted or incomplete data.

2.1.4 ITERATIVE CLOSED POINT

Iterative Closed Point (ICP), or iterative corresponding point, is described in multiple papers, such as[27] or[4]. It is an algorithm, which tries to fit a point cloud $\mathbf{A}$ on to another point cloud $\mathbf{B}$. Since it gets used as a step in many state-of-the-art pose estimations, such as FFB6D, which is introduced in [11] and described in Section 2.1.5, Iterative Closed Point (ICP) is explained in detail in this section.
ICP works by iteratively moving the centre of two point clouds to each other and rotating the point cloud in a way so that the average distance between the points of the two point clouds gets smaller with every iteration.
The goal is to find a rotation matrix $\hat{\mathbf{R}}$ and a translation vector $\hat{\mathbf{t}}$, that minimizes

the distance between points of $\mathbf{A}$ and $\mathbf{B}$. In other words, it tries to minimize the ADDS metric, which gets introduced in chapter 3, using $\hat{\mathbf{R}}$ and $\hat{\mathbf{t}}$, where the ground truth rotation matrix $\mathbf{R}$ and ground truth translation vector $\mathbf{t}$ are unknown.
In every iteration, first the centre of the two point clouds is calculated, meaning

$$\begin{aligned} \overline{\mathbf{c}}_{A_i} &= \frac{1}{|\mathcal{P}_\mathbf{A}|} \sum_{k=1}^{|\mathcal{P}_\mathbf{A}|} \mathbf{a}_k \\ &\text{and} \\ \overline{\mathbf{c}}_B &= \frac{1}{|\mathcal{P}_\mathbf{B}|} \sum_{k=1}^{|\mathcal{P}_\mathbf{B}|} \mathbf{b}_k, \end{aligned} \tag{2.2}$$

where $|\mathcal{P}_\mathbf{A}|$ denotes the amount of points of the point cloud $\mathbf{A}_i$, $\mathbf{a}_k$ is a column of the matrix $\mathbf{A}_i$ and represents a point of the point cloud $\mathbf{A}$. The same is done to point cloud $\mathbf{B}$, where $|\mathcal{P}_\mathbf{B}|$ denotes the amount of points in the point cloud and the column $\mathbf{b}_k$ represents one point. The matrix $\mathbf{A}_i$ is a $3 \times |\mathcal{P}_\mathbf{A}|$ matrix and $\mathbf{B}$ is of size $3 \times |\mathcal{P}_\mathbf{B}|$. The iteration index i is set to zero at the start and $\mathbf{A}_0 = \mathbf{A}$. $\overline{\mathbf{c}}_{\mathbf{A}_i}$ represents the centre of $\mathbf{A}_i$ and $\overline{\mathbf{c}}_\mathbf{B}$ the centre of $\mathbf{B}$.
Then $\mathbf{t}_i$ is found by

$$\mathbf{t}_i = \overline{\mathbf{c}}_\mathbf{B} - \overline{\mathbf{c}}_{A_i}. \tag{2.3}$$

Using $\overline{\mathbf{c}}_{\mathbf{A}_i}$ and $\overline{\mathbf{c}}_\mathbf{B}$ the centre points of $\mathbf{A}_i$ and $\mathbf{B}$ are translated to $\mathbf{0} \in \mathbb{R}^3$. This done by

$$\begin{aligned} \mathbf{Z} &= \mathbf{A}_i - \overline{\mathbf{C}}_{\mathbf{A}_i} \\ &\text{and} \\ \mathbf{Y} &= \mathbf{B} - \overline{\mathbf{C}}_\mathbf{B}, \end{aligned} \tag{2.4}$$

where $\overline{\mathbf{C}}_{\mathbf{A}_i}$ and $\overline{\mathbf{C}}_\mathbf{B}$ are obtained by multiplying $\overline{\mathbf{c}}_{\mathbf{A}_i}$ and $\overline{\mathbf{c}}_\mathbf{B}$ with $\mathbf{1}^T$, which is a vector filled with ones $\in \mathbb{R}^{|\mathcal{P}|}$. Meaning

$$\begin{aligned} \overline{\mathbf{C}}_{\mathbf{A}_i} &= \overline{\mathbf{c}}_{\mathbf{A}_i} \mathbf{1}^T \\ &\text{and} \\ \overline{\mathbf{C}}_\mathbf{B} &= \overline{\mathbf{c}}_\mathbf{B} \mathbf{1}^T. \end{aligned} \tag{2.5}$$

The matrix $\mathbf{T}_i$, which translate every point in $\mathbf{A}_i$ by $\mathbf{t}_i$, can be calculated with $\mathbf{T}_i = \mathbf{t}_i \mathbf{1}^T$.
By using $\mathbf{Z}$ and $\mathbf{Y}$, $\mathbf{R_i}$ can be found with

$$\mathbf{R_i} = \arg\min_{\mathbf{\Omega}} \|\mathbf{\Omega Z} - \mathbf{Y}\|_F^2, \tag{2.6}$$

where $\mathbf{\Omega}$ is a 3×3 rotation matrix and $\| \cdot \|_F$ denotes the Frobenius norm of a matrix.

So finding $\mathbf{R_i}$ is an orthogonal Procrustes problem, which can be solved by SVD-factorization, as described in [28]. This can be seen when rearranging the terms from equation 2.6,

$$\begin{aligned}\mathbf{R_i} &= \arg\min_{\mathbf{\Omega}} \|\mathbf{Z}\|_F^2 + \|\mathbf{Y}\|_F^2 - 2\langle \Omega\mathbf{Z}, \mathbf{B}\rangle_F \\ &= \arg\max_{\mathbf{\Omega}} \langle \Omega\mathbf{Z}, \mathbf{Y}\rangle_F \\ &= \arg\max_{\mathbf{\Omega}} \langle \Omega, \mathbf{Y}\mathbf{Z}^T\rangle_F \\ &= \arg\max_{\mathbf{\Omega}} \langle \Omega, \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\rangle_F \\ &= \arg\max_{\mathbf{\Omega}} \langle \mathbf{U}^T\Omega\mathbf{V}, \mathbf{\Sigma}\rangle_F \\ &= \arg\max_{\mathbf{\Omega}} \langle \mathbf{S}, \mathbf{\Sigma}\rangle_F.\end{aligned} \tag{2.7}$$

And since $\mathbf{S}$ is a product of orthogonal matrices, it is again orthogonal. Also, $\mathbf{\Sigma}$ is a diagonal matrix, meaning the inner product is maximized when $\mathbf{S}$ is equal to the identity matrix $\mathbf{I}$. From this is follows

$$\begin{aligned}\mathbf{I} &= \mathbf{U}^T\mathbf{R}_i\mathbf{V}, \\ \mathbf{R}_i &= \mathbf{U}\mathbf{V}^T,\end{aligned} \tag{2.8}$$

where $\mathbf{U}$ and $\mathbf{V}$ are obtained by the SVD-factorization of $\mathbf{Y}\mathbf{Z}^T$.
At the end of each iteration $\mathbf{A}_i$ is updated by applying the rotation and translation to it,

$$\mathbf{A}_{i+1} = \mathbf{R}_i\mathbf{A}_i + \mathbf{T}_i. \tag{2.9}$$

After a fixed amount of iterations or convergence is reached, the algorithm stops and $\hat{\mathbf{R}} = \mathbf{R}_i\mathbf{R}_{i-1}...\mathbf{R}_1\mathbf{R}_0$ and $\hat{\mathbf{t}} = \mathbf{R}_i\mathbf{t}_i + \mathbf{R}_{i-1}\mathbf{t}_{i-1} + ... + \mathbf{R}_1\mathbf{t}_1 + \mathbf{t}_0$.
Under the assumption that $\mathbf{B} \approx \mathbf{R}\mathbf{A} + \mathbf{T}$, i.e. the point cloud $\mathbf{B}$ is approximately a rotated and translated point cloud $\mathbf{A}$, the algorithm might converge to $\mathbf{R} \approx \hat{\mathbf{R}}$ and $\mathbf{t} \approx \hat{\mathbf{t}}$.
However, this can fail depending on the initial position. Since ICP is a greedy algorithm and does not know which points of $\mathbf{A}$ and $\mathbf{B}$ do correspond.

2.1.5 FULL FLOW BIDERECTIONAL FUSION NETWORK FOR 6D POSE ESTIMATION

Full Flow Biderectional Fusion Network for 6D Pose Estimation. (FFB6D) is a pose estimator introduced in [11]. It is currently one of the best performing pose estimators, achieving a score of 96.1% ADDS AUC on the YCB-Video benchmark and a score of 99.7% ADD Accuracy on the LineMOD benchmark, which is introduced in [13].

FFB6D is a pipeline that its core uses a deep learning neural networks for outputting eight key point per detected object. A downsampled point cloud is fitted to the key points using ICP, recovering the pose of the detected object. The pipeline also includes preprocessing, which for example sorts outs dead pixels (pixels with a value of zero) in the depth image, since less than 1% of depth image pixels are actually used by the neural network. The network uses a full flow biderectional fusion approach of RGB and depth data, i.e. after each convolution the RGB and depth data are fused again, in a point to pixel and pixel to point manner. This improves performance by about 0.5% ADDS AUC, when compared with its predecessor PVN3D, which is introduced in [12].
The weights provided by [11], were trained with some disturbances in the RGB image, but not in the depth image. The disturbances in the RGB image are blurring, sharpening, changes in brightness and colour shifts.

2.2 FINDING SYMMETRY AXIS OF OBJECTS

Some evaluation metrics, which are introduced in Chapter 3, require every correct pose of an object. If an object has multiple correct poses, but the ground truth from the dataset only provides one correct pose, the additional correct poses can be found by rotating the object around its symmetric axes.
It is possible to find the symmetric axes of an object, by applying Principal component analysis (PCA) to its point cloud. This process is explained in [2, 10, 22]. To summarize, PCA is applied to the point cloud, where the individual points are treated as data points and the resulting components are the symmetric axes. This method has some limitations, the amount of symmetric axes must be known beforehand, since using PCA on 3D data will always find three components. The component with the largest eigenvalue is the first symmetry axis, the component with the second largest the second symmetry axis and so on. This also means at most 3 symmetry axes can be found using PCA. Also, the symmetries must be clearly visible in the point cloud. Because the PCA components represent the axes, where spread is maximized, which normally closely aligns with the symmetric axes, if they exist.
Since point clouds do not perfectly cover the object surface, the exact symmetry axes can not be found using PCA, however with high density point clouds the approximation is close enough for practical applications.

2.2.1 PRINCIPAL COMPONENT ANALYSIS

For completeness, how PCA is applied to 3D point clouds is explained in this section, as described in [19]. Given a point cloud in the form of a matrix $\mathbf{A}^T$ of

size $|\mathcal{P}_\mathbf{A}| \times 3$ PCA is applied as follows. The point cloud is translated to $\mathbf{0} \in \mathcal{R}^3$ by subtracting the centre $\overline{\mathbf{C}}_\mathbf{A}$,

$$\mathbf{Z}^T = \mathbf{A}^T - \overline{\mathbf{C}}_\mathbf{A}^T. \tag{2.10}$$

Then the covariance matrix $\mathbf{C}$ is calculated,

$$\mathbf{C} = \frac{1}{|\mathcal{P}_\mathbf{A}|}\mathbf{Z}\mathbf{Z}^T. \tag{2.11}$$

The eigenvalues of the covariance matrix $\mathbf{C}$ are denoted as λ_1, λ_2 and λ_3 in descending order. The corresponding eigenvectors are can be interpreted as the most likely symmetric axes $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$.

2.3 CONVERTING DEPTH IMAGES TO POINT CLOUDS

To calculate the 3D point cloud from a depth image the intrinsic camera matrix $\mathbf{K}$ is needed. The matrix $\mathbf{K}$ contains the parameters f_x and f_y, which represent the focal length, as well as the optical centre coordinates c_x and c_y. The parameters are arranged into $\mathbf{K}$ as follows,

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.12}$$

A singe pixel is given by pixel coordinates p_x, p_y and the value of the pixel d, given in meters. To project a pixel of the depth image to a point in 3D space, given by the coordinates x, y and z, the following mapping is applied:

$$\begin{aligned} z &= \frac{d}{\sqrt{1 + \frac{(p_x - c_x)}{f_x} + \frac{(p_y - c_y)}{f_y}}} \\ y &= \frac{(p_y - c_y)z}{f_y} \\ x &= \frac{(p_x - c_x)z}{f_x}. \end{aligned} \tag{2.13}$$

This mapping is applied to every pixel in the depth image. This mapping is based on [9, pp. 153-155] and assumes a pinhole camera model, which is also introduced in [9, pp. 153-162].

2.4 PROJECTING FROM 3D SPACE TO 2D IMAGE SPACE

To project a 3D point to the 2D image plane the following mapping can be used, assuming the 3D point is already in the camera coordinate system,

$$\begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} = \frac{1}{z} K \begin{bmatrix} x \\ y \\ z \end{bmatrix} . \tag{2.14}$$

$\mathbf{K}$ is the intrinsic camera, (x, y, z) are the coordinates of the 3D point in the camera coordinate system and p_x and p_y are the resulting 2D image pixels. This mapping is taken from [9, pp. 153-155].

3 Methodology

In this chapter current evaluation methods from the literature and a new novel evaluation method are introduced and discussed.
The evaluation process can be divided into two phases. Firstly, an error metric is calculated for every estimation (including false negatives and false positives). The lower the metric, the better the estimation. A metric of zero means the estimated pose is perfect and is equal to ground truth. In practise, this almost never happens due to noise and numerical instability. Even the best pose estimators can not be expected to output perfect estimations. Metrics play a central role in evaluating pose estimators and analysing them under disturbances, because some metrics are more sensitive to different errors than others. 6D pose estimation is a combination of detecting the objects, estimating their position and their rotation.
Secondly, on the basis of the metrics, a score is then calculated. A metrics quantifies how good a single estimation is and scores how good the entire pose estimator is. The scores are calculated over a benchmark dataset, like YCB-Video benchmark which is introduced in [36], to make the performance of different pose estimators comparable. The higher the score, the better the pose estimator. It is desirable that a score is between zero and one, so that it can be interpreted as a percentage, because then a score of one means the pose estimator is perfect, i.e. every estimation has an error metric of 0. A score combined with one or multiple metrics establishes an evaluation method.
For the task of 6D pose estimation, there are three error sources: rotation, translation and detections. If a detection is right or wrong, i.e. is the detected object in the to be evaluated frame is often evaluated in the score calculation, while the metrics cover translation and rotation errors.
The choice of evaluation method is important when comparing pose estimators, since some evaluation simply ignore entire error sources (like false detections for example), or place little weight on them. Depending on the use case putting a low weight on an error source might be desired, since the low weighted error source does not matter for the task at hand. For the case of finding the influence of different disturbances, every error source is important. Because a disturbance

might only affect one error source, if this error source is then ignored in the evaluation, the effect of the disturbance goes unnoticed.
To discuss the different metrics in this Chapter characteristic of a good metric are now defined. A good metric should be sensitive to both translation and rotation. It should also be in the same bounds for different objects, and deliver similar results for different objects. If some objects have a higher metric upper bound, these objects get punished more for the same error in their pose estimation. Since the task is to evaluate the pose and not the object, it should be expected that two similar objects give the same error metric for the same pose estimation. Similar in this context means, that they have the same amount of correct poses, since symmetric objects for example have two correct poses, which some metrics ignore. Generally, metrics sort objects into two categories, objects with a single correct pose and objects with multiple correct poses. An object with multiple correct pose has at least one symmetric or rotation invariant axis.
Furthermore, the metric should weigh the rotation and translation similarly. For example, if 1° error in the rotation gives the same error as 10 meters in translation, the translation can be almost ignored entirely by the pose estimator. Because none of the metrics in literature fulfil the criteria a new novel metric is introduced in section 3.2, alongside a new evaluation score which treats the issue of false detections.

3.1 LITERATURE REVIEW ON EVALUATION METHODS

In this Section the current evaluation methods are introduced.

3.1.1 AVERAGE DISTANCE OF MODEL POINTS METRIC

The Average Distance of model points (ADD) metric e_{ADD} is introduced in [13] and defined as

$$e_{\mathsf{ADD}} = \frac{1}{|\mathcal{P}_s|} \sum_{\mathbf{p} \in \mathcal{P}_s} \|(\mathbf{R}\mathbf{p} + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{p} + \hat{\mathbf{t}})\|_2. \tag{3.1}$$

Here, $\mathcal{P}_s$ represents the point cloud of an object with a singular correct pose, and $|\mathcal{P}_s|$ its cardinality. $\mathbf{R}$ and $\mathbf{t}$ are the ground truth rotation matrix and translation vector for the object, respectively, with $\hat{\mathbf{R}}$ and $\hat{\mathbf{t}}$ representing the estimated counterparts. This metric effectively measures the average distance between corresponding point pairs from the estimated pose to the ground truth. In the context of 6D pose estimation error metrics corresponding points are defined a pair of points where one is in the ground truth point cloud and the other in the estimated point cloud $\mathcal{P}_{\mathsf{est}}$ and these must be same point in their respective point cloud, i.e. the points have the same coordinates if the same rotation and

translation is applied to both point clouds. A visualization of the ADD metric can be found in Figure 3.1.

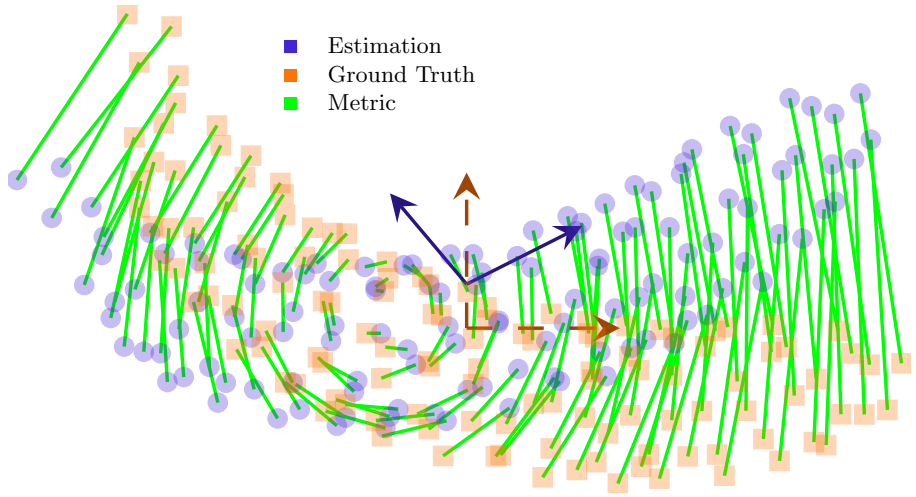

Figure 3.1: Visualization of the ADD metric on a banana from the YCB-Video dataset. The blue point cloud represents the pose estimation; the orange point cloud represents the ground truth. The pose estimation is off by a rotation of 30° on the Z-axis and a translation of 1 cm on the X-axis. For better visualization, the object's local X and Y-axis are also drawn at the centre of each point cloud. The green lines represent the ADD metric, with their average length illustrating the error metric.

3.1.2 AVERAGE DISTANCE OF MODEL POINTS FOR SYMMETRIC OBJECTS METRIC

Average Distance of model points for Symmertic Objects (ADDS) e_{ADDS} is introduced in [13] and defined as

$$e_{\text{ADDS}} = \frac{1}{|\mathcal{P}_m|} \sum_{\mathbf{p}_1 \in \mathcal{P}_m} \min_{\mathbf{p}_2 \in \mathcal{P}_m} \|(\mathbf{R}\mathbf{p}_1 + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{p}_2 + \hat{\mathbf{t}})\|_2. \tag{3.2}$$

Here, $\mathcal{P}_m$ is the point cloud of an object with one or multiple correct poses. This metric measures the average minimal distance between two points in the estimated pose and the ground truth pose. It can be interpreted as a metric for surface overlap. A visualisation can be found in Figure 3.2.

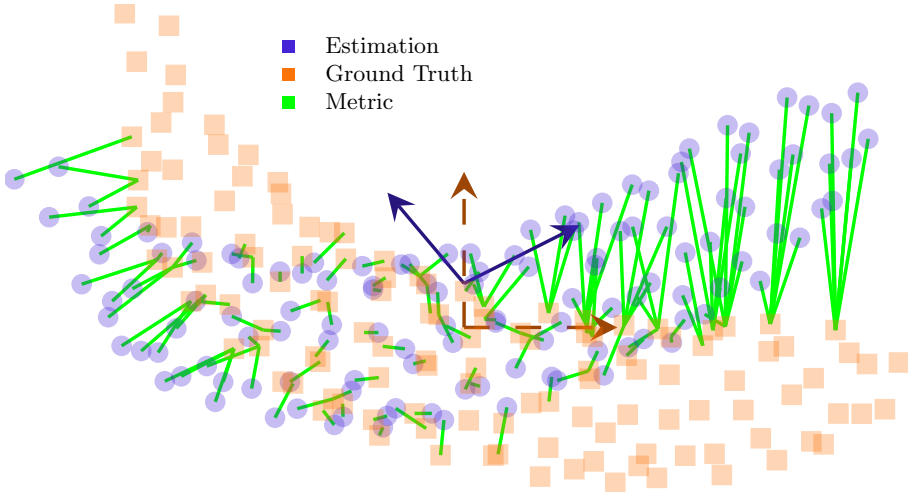

Figure 3.2: A visualization of the ADDS metric on a banana, from the YCB-Video dataset. The blue point cloud is the pose estimation, the orange point cloud is the ground truth. The pose estimation is off by a rotation of 30° on the Z-axis and a translation of 1 cm on the X-axis. For better visualization, the object's local X and Y-axis are also drawn at the center of each point cloud. The green lines represent the ADDS metric, the average length of these lines is the resulting metric.

3.1.3 TRANSLATION ERROR METRIC

The translation error (TE) e_t is defined as

$$e_t = \|\mathbf{t} - \hat{\mathbf{t}}\|_2, \tag{3.3}$$

in accordance with [14].

3.1.4 ROTATION ERROR METRIC

The rotation error (RE) tries to measure the difference between two rotation matrices. An overview over different definitions of rotation error metrics can be found in [17]. Since these definitions are functional equivalent with each other and the definition in [14], the rotation error e_r is defined as

$$e_r = \|\mathbf{I} - \mathbf{R}\hat{\mathbf{R}}^T\|_F, \tag{3.4}$$

in this thesis, for its easy geometric interpretation. Here $\|\cdot\|_F$ denotes the Frobenius norm. This definition of $e_r \in [0, 2\sqrt{2}]$.

3.1.5 COMPLEMENT OVER UNION METRIC

TheComplement over Union (CoU) is defined as

$$e_{\text{CoU2D}} = 1 - \frac{\text{Area}(\hat{A} \cap A)}{\text{Area}(\hat{A} \cup A)}, \tag{3.5}$$

in accordance with [14]. Here A is the 2D bounding box (or segmentation mask) of the ground truth object and $\hat{A}$ is the 2D bounding box (or segmentation mask) of the object after the estimated translation and rotation is applied.
Usally this metric is used in 2D object detection, it is possible to project the bounding boxes or segmentation mask from 3D space to 2D space, as is explained in Section 2.4. In practise, it is better to use the volume of the 3D bounding boxes, or segmented objects, as is done in [37]. Resulting in

$$e_{\text{CoU3D}} = 1 - \frac{\text{Volume}(\hat{B} \cap B)}{\text{Volume}(\hat{B} \cup B)}, \tag{3.6}$$

where B and $\hat{B}$ are 3D bounding boxes. Both e_{CoU2D} and $e_{\text{CoU3D}} \in [0, 1]$

3.1.6 AVERAGE CORRESPONDING POINT DISTANCE METRIC

The Average Corresponding Point Distance (ACPD) is introduced in [14] and is an extension of the ADD metric. It is defined as

$$e_{\text{ACPD}} = \min_{(\mathbf{R},\mathbf{t}) \in \mathcal{Q}} \frac{1}{|\mathcal{P}_m|} \sum_{\mathbf{p} \in \mathcal{P}_m} \|(\mathbf{R}\mathbf{p} + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{p} + \hat{\mathbf{t}})\|_2. \tag{3.7}$$

The set $\mathcal{Q}$ contains all correct poses of an object. How this set is obtained is not mentioned in [14].[1] Unlike the ADD metric, the ACPD is not depended on how many correct poses an object has.

3.1.7 MAXIMUM CORRESPONDING POINT DISTANCE METRIC

The Maximum Corresponding Point Distance (MCPD) is similar to the ACPD, with the only difference being that the maximum is taken instead of the average. It was also introduced in [14] and is defined as

$$e_{\text{MCPD}} = \min_{(\mathbf{R},\mathbf{t}) \in \mathcal{Q}} \max_{\mathbf{p} \in \mathcal{P}_m} \|(\mathbf{R}\mathbf{p} + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{p} + \hat{\mathbf{t}})\|_2. \tag{3.8}$$

[1] This set can be obtained by adding additional rotations to the ground truth, how this is done is explained in Section 2.2

3.1.8 VISIBLE SURFACE DISCREPANCY METRIC

Since [14] does not mentation a way to obtain the set $\mathcal{Q}$ it also introduces the Visible Surface Discrepancy (VSD), which only takes the visible surface of the object under consideration into account. The visible surface is defined as the surface visible from the sensor setup, which is in most cases is just the RGB camera. Hence the set $\mathcal{V}$ contains all pixels from the object under consideration, which can be seen from the ground truth object, and the set $\hat{\mathcal{V}}$ contains all visible pixel of the estimation. The Visible Surface Discrepancy (VSD) is defined as

$$e_{\text{VSD}} = \frac{1}{|\mathcal{V} \cup \hat{\mathcal{V}}|} \sum_{p \in \mathcal{V} \cup \hat{\mathcal{V}}} c(p, \lambda). \tag{3.9}$$

Here the cost matching function $c(p, \lambda)$ is defined as

$$c(p, \lambda) = \left\{ \begin{array}{ll} d/\lambda & \text{if } p \in \mathcal{V} \cap \hat{\mathcal{V}} \text{ and } d < \lambda \\ 1 & \text{otherwise} \end{array} \right\}, \tag{3.10}$$

where λ is the misalignment tolerance, which allows the maximal range of d and d is the distance between the two pixels p from the ground truth object and the estimation in 3D space. The position of a pixel in 3D space can be found using the intrinsic camera matrix and the depth image, which is explained in Section 2.3.
The VSD $\in [0, 1]$ since the function $c(p, \lambda)$ is also $\in [0, 1]$.

3.1.9 ACCURACY SCORE

The accuracy s_a is the percentage of correct poses. Correct poses are those for which the metric is below a threshold τ.

$$s_a = \frac{1}{|\mathcal{O}_g|} \sum_{l=0}^{|\mathcal{O}_g|} \chi(e_l < \tau), \tag{3.11}$$

where χ is the indicator function, which returns 1 when $e_l < \tau$ and 0 otherwise. The set $\mathcal{O}_g$ contains all ground truth objects in the evaluation dataset and e_l is the error metric for an object $\in \mathcal{O}_g$.

3.1.10 MEAN RECALL SCORE

The mean recall (MR) s_{MR} is defined as the average percentage of correctly classified poses for a set of thresholds $\mathcal{T}$,

$$s_{MR} = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} h(\tau), \tag{3.12}$$

$h(\tau)$ returns the percentage of correctly classified poses for the threshold τ and can be interpreted as the accuracy s_a depending on threshold τ. This definition was taken from [14]. Since this score is averaging over percentages it is $\in [0, 1]$.

3.1.11 AREA UNDER CURVE SCORE

The area under curve (AUC) s_{AUC} is defined as the integral of the proportion of correctly classified poses, over varying thresholds. Since it currently is the gold standard score, in combination with ADD and ADDS it used by [36, 11, 29, 32, 23, 33, 34, 30] and is defined as

$$s_{AUC} = \int_0^{\gamma} h(\tau), \tag{3.13}$$

where γ is the maximum threshold at which the evaluation starts. A visual representation can be seen in Figure 3.3. This AUC is closely related to the MR, since the AUC is just the mean recall (MR) with infinite thresholds. In practise the thresholds would be set exactly after the metric values, i.e. the AUC gets numerically calculated like the MR with perfect threshold placement to separate all error metric values.

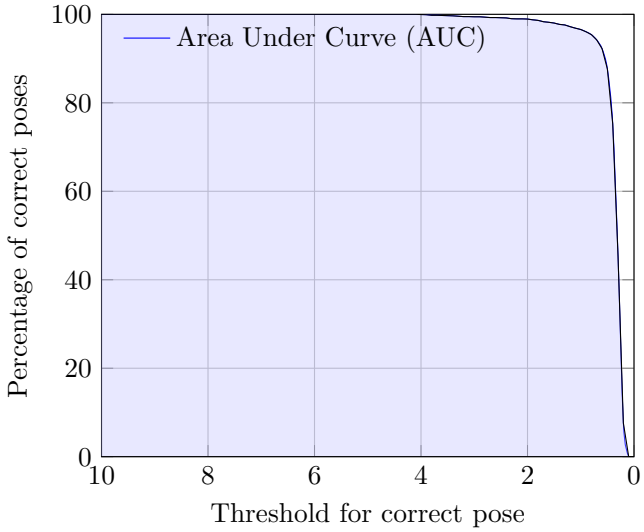

Figure 3.3: An example for the AUC. This AUC is from FFB6D on the YCB-Video benchmark, using the ADDS metric. The thresholds from 10 cm to 0 cm are plotted on the X-axis, the percentage of correct poses is plotted on the Y-axis. The area under the line is the AUC, in this example it is 96.13%.

3.1.12 INTERSECTION OVER UNION

The Intersection over union (IoU) s_{IoU3D} is the complement of the compliment over union,

$$s_{\text{IoU3D}} = \frac{\text{Volume}(\hat{B} \cap B)}{\text{Volume}(\hat{B} \cup B)}. \tag{3.14}$$

As for the CoU, instead of the volume, the area of the 2D projection could also be used. This definition was taken from [14] and is only applicable to one object. When calculating theComplement over Union (CoU) for an entire dataset, the score is averaged over all objects. This score can be interpreted as the percentage of overlapping area or volume, between the estimation and ground truth.

3.2 PROPOSED NOVEL EVALUATION METHODS

In this Section the new novel evaluation method is introduced. It introduces a new error metric, which can be viewed as an extension and combination of the rotation and translation error and a new evaluation score. The error metric was developed to treat every kind of object the same, independent of the amount of correct poses, while being able to weigh the importance of rotation and translation. The proposed evaluation score is able to treat the issue of false detections and no longer treats poses as a binary problem, i.e. correct or incorrect.

3.2.1 SYMMETRIC ROTATION ERROR METRIC

The rotation error can be extended for symmetric objects, for this the symmetry axis $\mathbf{u}_i$ and the corresponding correct angles $\theta_{i,j} \in \mathcal{B}_i$ must be known. The set $\mathcal{B}_i$ contains the symmetry angles for a symmetry axis $\mathbf{u}_i$. For example, for an object with a fourfold symmetry around the X-axis: $\mathbf{u}_1 = (1, 0, 0)$ and $\mathcal{B}_{i1} = \{0°, 90°, 180°, 270°\}$.
The main idea is to add the additional rotations to the ground truth, which also lead to a correct pose, since objects with multiple correct poses are either symmetric or rotation invariant on one axis, i.e. their correct poses only differ in rotation. Only the combination of rotations, that leads to the smallest error is used for the error metric. To achieve this, we need the rotation matrix $\tilde{\mathbf{R}}_{\mathbf{u},\theta}$, which contains a rotation around the axis $\mathbf{u} = (u_x, u_y, u_z)$ with the angle θ. This matrix is defined as [21, 20]:

$$\tilde{\mathbf{R}}_{\mathbf{u},\theta} = \begin{pmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_y u_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_z u_x(1-\cos\theta) - u_y\sin\theta & u_z u_y(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{pmatrix}. \tag{3.15}$$

However, since the symmetry axes are given in world coordinate, they do not align with the local object coordinate after one rotation. To make the axis $\mathbf{u}_i$ align with the object coordinate system again, all previous rotations must be applied. For example, after the ground truth was rotated by $90°$ on the X-axis $(1, 0, 0)$, the second symmetry axis $(0, 1, 0)$ is now $(0, 0, 1)$.
This leads to the set $\mathcal{A}$, which contains all symmetry axes of an object with the previous rotations applied,

$$\begin{aligned}\mathcal{A} &= \left\{ \mathbf{R}\mathbf{u}_1, \tilde{\mathbf{R}}_{\mathbf{R}\mathbf{u_1},\theta_{\mathbf{1,j}}}\mathbf{R}\mathbf{u}_2, \tilde{\mathbf{R}}_{\tilde{\mathbf{R}}_{\mathbf{R}\mathbf{u_1},\theta_{\mathbf{1,j}}}\mathbf{R}\mathbf{u_2},\theta_{\mathbf{2,j}}}\tilde{\mathbf{R}}_{\mathbf{R}\mathbf{u_1},\theta_{\mathbf{1,j}}}\mathbf{R}\mathbf{u}_3, \dots \right\} \\ &= \{\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}, \dots\}.\end{aligned} \tag{3.16}$$

Which leads to the set of all possible correct rotations around symmetric axes $\mathcal{R}$ for an object after the ground truth rotation is applied, being defined as,

$$\mathcal{R} = \left\{ \prod_{\mathbf{v}_i \in \mathcal{A}} \tilde{\mathbf{R}}_{\mathbf{v}_i,\theta_j} \mid \theta_j \in \mathcal{B}_i \right\}. \tag{3.17}$$

Resulting in the symmetric rotation error e_{RSYM} being defined as

$$e_{\mathsf{RSYM}} = \min_{\tilde{\mathbf{R}} \in \mathcal{R}} (\|\mathbf{I} - \tilde{\mathbf{R}}\mathbf{R}\hat{\mathbf{R}}^T\|_F). \tag{3.18}$$

3.2.2 ROTATION INVARIANT ROTATION ERROR METRIC

The rotation error e_{r} can also be extended to handle objects with rotation invariance. Since it has been proven in [7], that an object can only be rotation invariant on one axis, or else it would be a ball i.e. it is rotation invariant on all axes, only the case of a single rotation invariant axis needs to be considered.
That means, that the rotation invariant rotation error e_{RINVAR} can be viewed as an optimisation problem in one dimension,

$$e_{\mathsf{RINVAR}} = \min_{\alpha} (\|\mathbf{I} - \tilde{\mathbf{R}}_{\mathbf{w},\alpha}\mathbf{R}\hat{\mathbf{R}}^T\|_F). \tag{3.19}$$

where $\mathbf{w}$ is the rotation invariant axis, with previous rotations applied and α is the angle which minimizes the error e_{RINVAR} by rotating around $\mathbf{w}$.
Since α is the only optimisation parameter and represents a rotation around one axis, the optimisation function is rather simple. The function to optimize is of periodic nature and every local minimum is also a global minimum, that rotation error under rotation on one axis behaves this way can be seen in [14].

3.2.3 MULTI ROTATION ERROR METIC

The multi rotation error (MRE) is a combination of the symmetric rotation error e_{RSYM} and the rotation invariant rotation error e_{RINVAR}. The main advantage to the MRE is that it can be applied to any object, no matter how many correct poses the object has. The MRE e_{R} is defined as

$$e_{\mathsf{R}} = \min_{\tilde{\mathbf{R}} \in \mathcal{R}} (\min_{\alpha} (\| I - \tilde{\mathbf{R}}_{\mathbf{w},\alpha} \tilde{\mathbf{R}} \mathbf{R} \hat{\mathbf{R}}^T \|_F)). \tag{3.20}$$

The set $\mathcal{R}$ contains all possible correct rotations around the symmetric axes of an object, after the ground truth rotation is applied. If the object under consideration has no symmetries $\mathcal{R} = \{\mathbf{I}\}$. $\mathbf{w}$ is the rotation invariant axis of an object with the previous rotations (ground truth and symmetric rotations) applied. If an object has no rotation invariant axis, α is simply set to zero.

3.2.4 MULTI ROTATION TRANSLATION ERROR METRIC

Since the MRE e_{R} only evaluates the rotation and not the translation of an estimated pose, it can be combined with the translation error e_{t}, resulting in the multi rotation translation error (MRTE) e_{MRTE}, which is defined as

$$e_{\mathsf{MRTE}} = f(e_{\mathsf{R}}) + g(e_{\mathsf{t}}). \tag{3.21}$$

Where f and g scaling functions for e_{R} and e_{t}. It is recommended to scale because, $e_{\mathsf{R}} \in [0, 2\sqrt{2}]$ and $e_{\mathsf{t}} \in [0, \infty)$. Also, by applying scaling, the importance of rotation or translation can be given for the specific task.
Some default scaling would be to linearly scale e_{R} to be in the interval of $[0, 1]$, $f(x) = \frac{x}{2\sqrt{2}}$. Since e_{t} has no upper bound, it can be cut off after a threshold, after which the estimation is unusable. This threshold is called the usability threshold β, resulting in $g(x) = \min(\frac{x}{\beta}, \beta)$. For the ADD and ADDS AUC the maximum threshold γ is often set to 10 cm, as proposed in [36], the same value can be used for β to make the metrics comparable.

3.2.5 AVERAGE INVERSE MULTI ROTATION TRANSLATION ERROR SCORE

The Average Inverse Multi Rotation Translation Error Score (AIMRTES) s_{AIMRTES} was designed to be used with the MRTE, because as the name suggests it is defined as the average inverse of the error. However, to also cover the aspect of detection and giving it bounds in $[0, 1]$, one is added to the denominator. Resulting in

$$s_{AIMRTES} = \frac{1}{|\mathcal{O}_d|} \sum_{k=0}^{|\mathcal{O}_d|} \frac{1}{e_k + 1}, \tag{3.22}$$

where $\mathcal{O}_d$ is the union of all detected objects and all correct objects in the evaluation dataset and e_k is the error e for an object $\in \mathcal{O}_d$. For objects, which were not detected, the error e_k is set to ∞, meaning zero is added to the sum over all objects. The score S lies in the interval $[0, 1]$.[2]
Other error functions can also be used, the upper bound of the error function determines how much the correct detection of an object gets weighted into the score.

3.3 THEORETICAL DISCUSSION

In this section, the advantages and disadvantages of the metrics and scores are discussed. The selection of an evaluation method is not trivial, since the metrics and scores are measuring different aspects of the estimated poses. In [16] an overview of the entire 6D pose estimation progress is given, since different evaluation methods were used in some approaches it makes the performance hard to compare to other pose estimators.
There have already been discussions focused specifically on the evaluation methods for 6D pose estimation in [14], where new scores and metrics are introduced, and the conclusion is drawn that different use cases focus on different aspects of the pose estimation. According to [14] object grasping in the field of robotics is mainly focused on surface overlap.
In [14] the ACPD, MCPD and the VSD are introduced. However, currently the ADD and ADDS metrics are still the most commonly used metrics, which is probably due to their easy interpretation and the ACPD and MCPD not addressing every problem of the ADD metric.
To better understand ADD, ADDS and the proposed MRTE, the framework of [14] was followed to provide some visualizations on how the metrics behave. In [14] an error, like rotation, is isolated and the behaviour of the different metrics is shown. This was only done for rotation from $0°$ to $360°$ for a coffee mug, which is not sufficient to see the behaviour of all metrics.
In the following visualisations of the metrics all objects from the YCB-Video dataset are used, as well as different error sources.

[2] Note that e_R and e_t, do not need to be scaled with an upper bound of one. The range can be viewed as weighting the score for rotation, translation, and object detection. If both the rotation and translation error are scaled with an upper bound of one, rotation, translation, and detection all contribute the same amount to the final score. It is also possible to just iterate over all correct objects in the evaluation dataset, like it is done when using ADD and ADDS AUC. An implementation can be found under `https://github.com/D-Doge/metric_pose_estimation`.

3.3.1 METRICS UNDER TRANSLATION

In Figure 3.4 the behaviour of the different error metrics under translation is depicted. The MRTE, ADD, MCPD and ACPD all behave the same. They all have a linear relationship with the translation error, in fact they are equal to the translation. The MRTE only uses the translation error in this case, which just gives the distance of the translation. The same is true for the ADD metric, since it gives the average distance of all points from the ground truth to the estimation. In this case, every point has the same distance, i.e. the MCPD and the ACPD are equal to the ADD metric in this case, because the correct rotation ($\hat{\mathbf{R}}$ is equal to the unit matrix) is used, for every metric.
The results from the ADDS metric are scattered, every object has a visibly different line. This is due to the fact, that the ADDS metrics does not use corresponding points of the ground truth and estimation, but simply the ones with the minimal distance. This means, that when the ground truth and estimated point clouds still overlap, the error grows slowly. Only when the point clouds start to move further apart, the ADDS metric starts to grow linearly.
From about 0 to 5 cm the objects still overlap, however, even after that the ADDS metric still grows at different speeds for every object. This makes the ADDS heavily dependent on the point cloud under translation. At 10 cm translation the ADDS metric has values ranging from about 4.1 cm to 9.1 cm, which is a difference of over 200%. Also when using the ADDS AUC with a starting threshold of 10 cm, as proposed in [36], only after about 18 cm of translation the AUC would be zero. With 5 cm of translation, the AUC would be over 70%. This suggests, that the ADDS is insensitive to translation, when compared to the other metrics.

3.3.2 RESULTS UNDER ROTATION

For the error metrics under rotation, three different kind of objects need to be considered. The first kind of objects are those with a single correct pose, an example would be a banana. The second kind of objects are those with symmetries. There are different kinds of symmetries (twofold, threefold etc.) that need to be considered, but for the following visualization only twofold symmetries were used, since in the YCB-Video dataset only the woodblock has a different kind of symmetry along one axis (fourfold). The third kind of objects are those with a rotation invariant axis, in the YCB-Video dataset only the bowl is considered rotation invariant.
Because the MRTE has vastly different upper bounds than the point cloud based metrics multiple plots are provided.

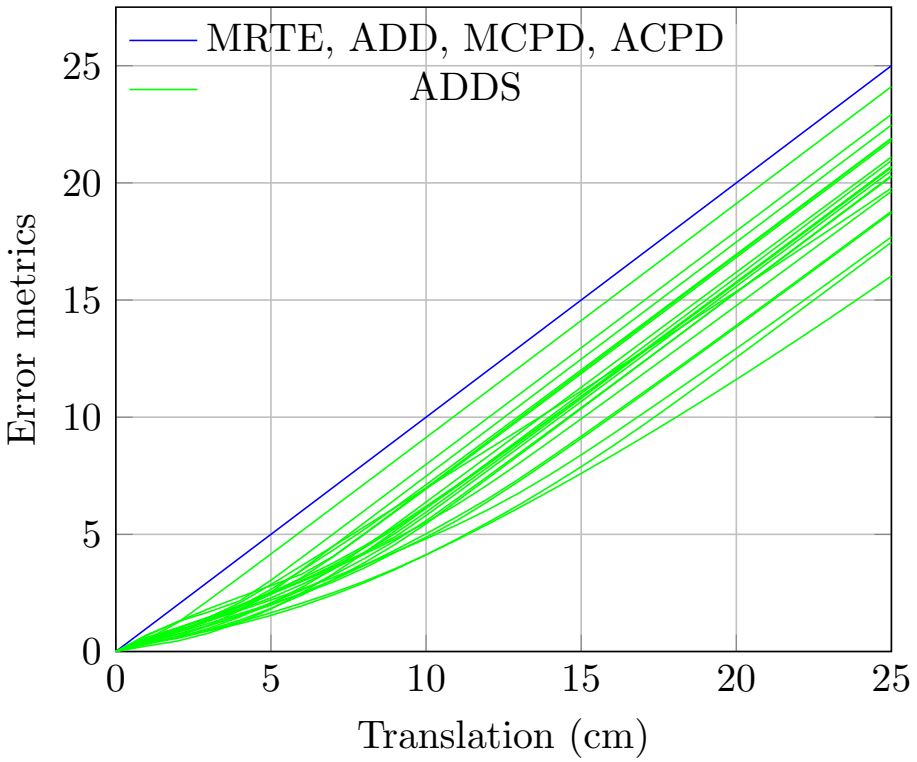

Figure 3.4: The error metrics under translation, for every object in the YCB-dataset. On the X-axis the error (translation on the X-axis) is plotted and on the Y-axis the resulting metric. Every metric can be interpreted as cm. The MRTE can only interpreted as cm when no scaling to the translation error is applied. Each line represents one object evaluated with one metric. The MRTE, ADD, MCPD and ACPD are represented by the blue lines, because they are all equal, while the ADDS is represented by the green lines and behaves different for every object.

MRTE UNDER ROTATION

In Figure 3.5 the MRTE metric under rotation, for all three kinds of objects can be seen. Objects with one correct pose were rotated around the X-axis. For this kind of objects, the MRTE behaves the same as the rotation error e_r and produces the same error values for every object in this category. Since every object with one correct pose gets treated the same, they have a consistent maximum value at π radians. Also, the MRTE behaves linear under rotation.
For twofold symmetric objects, the MRTE takes the minimum of two rotation errors e_r, where one is shifted by π radians. This means that the MRTE has two minima and two maxima, reflecting the two correct poses. Both minima have the exact same value of zero, and both maxima the exact value of $\pi/2$. The minima are at the correct rotations and the maxima in the middle of them. It is obvious, that this pattern would repeat for objects with more symmetries along the axis. For example, a fourfold symmetry would have four maxima and four minima. The transition between maxima and minima is linear. Objects with more symmetries have a lower upper bound, this could be counteracted by scaling the error on this axis. This is easy to do since the upper bound can be predicted by dividing π by the amount of symmetries. But this is not implemented into the MRTE, since otherwise objects with a high amount of symmetries would have a faster growing error than objects with no symmetry. For example, an object with 180 symmetries along a single axis would have its maximum error at every odd degree, which means even a one degree error in the pose estimation leads to a metric error value of π, when the rotation error is scaled for symmetric objects as described above. Because scaling the rotation error as described above leads to an extremly senstive metric, the MRTE does not scale the rotation error for symmetries, which implies that the rotation on symmetric axis is less important.
For objects with a single rotation invariant axis, the MRTE behaves as expected and produces values near zero for every possible rotation along the invariant axis. The values are not zero exactly due to numerical errors.

POINT CLOUD BASED METRICS UNDER ROTATION

In Figure 3.6 the ADD, ADDS ACPD and the MCPD are plotted, under rotation for objects with one correct pose. All objects with a single correct pose from the YCB-Video dataset were used. All objects were rotated around the X-axis from $0°$ to $360°$. It can be seen that the ADD, ADDS, ACPD and MCPD all have different upper bounds for different objects. Since the ACPD and MCPD are variations of the ADD metric they behave similarly. The ACPD behaves exactly like the ADD metric, which is expected, since the objects have only one correct pose, i.e. the ACPD is equal to the ADD metric. The MCPD only considers the point which is

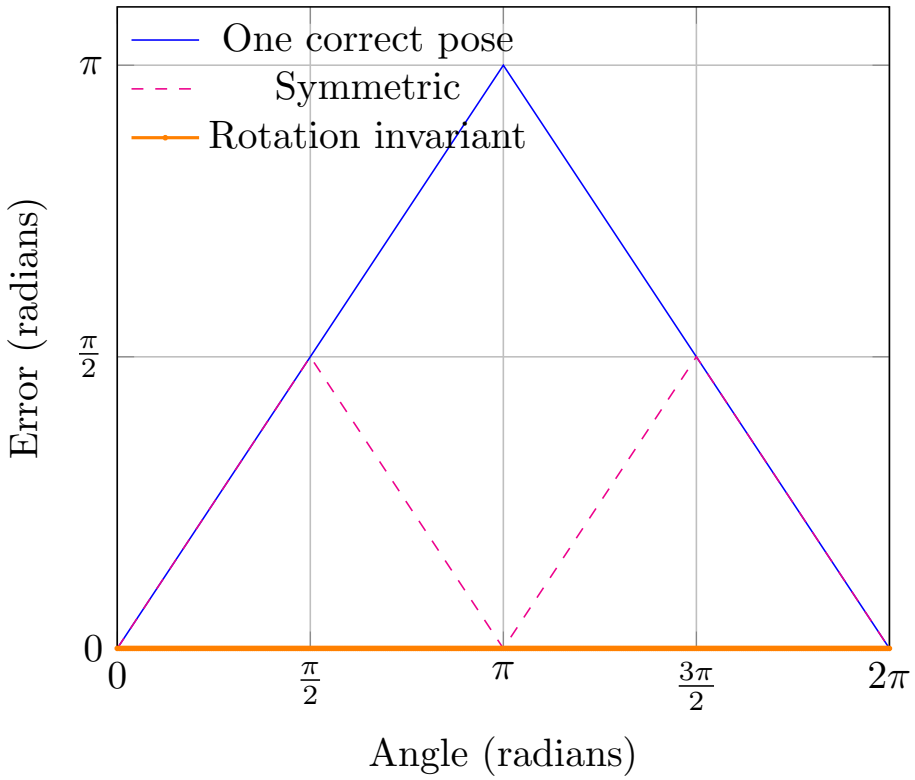

Figure 3.5: The MRTE metric under rotation for all three kinds of objects. Every object from the YCB-Video dataset was used. The blue lines represent objects with one correct rotation, the dashed magenta line objects with a twofold symmetry and the thick orange line objects with a rotation invariant axis. Objects with one correct pose were rotated from 0° to 360°, around the X-axis. Symmetric objects were rotated around their symmetric axis and rotation invariant object around their rotation invariant axis. The rotation is plotted on the X-axis of the plot, and the resulting error metric on the Y-axis. In can be seen that the MRTE gives the same value for all objects of the same kind.

furthest away from its corresponding point, which in practise results in ACPD with larger values. The results of the ADD, ACPD and MCP all have a parabolic shape, meaning the gradient is smaller around the maximum. When comparing the maximum error values of different objects, the spread is large. The upper bounds of the ADD and ACPD are between about 4.9 cm and 17.7 cm, which is a difference of more than 300%. Such a large difference in upper bounds makes it difficult to assign a threshold for score calculations. Following the difference in upper bounds, the gradients are also very different for different objects, which is not ideal when the metrics are used as loss functions, since the optimizer is more focused on objects with larger gradients. The difference in upper bounds and gradients can also be seen for the MCPD. The absolute values for the upper bounds are larger, between about 7 cm and 25 cm, but the increase is also a bit larger than 300%. The ADDS metric behaves differently, when compared to the ADD, ACPD and MCPD. While the ADD, ACPD and MCPD have a single maximum at π radians and a minimum at 0 radians, the ADDS metric sometimes has two maximums at around $\pi/2$ radians and $3\pi/2$, with a second minimum at π. This implies symmetry around the X-axis (the rotation axis), which is not the case. For some objects the ADDS error is close to 0 cm and does not change much, for any rotation, which implies rotation invariance around the X-axis. Considering how the ADDS metric is calculated, it becomes obvious that the ADDS metric does not actually measure rotation or translation errors, but the average surface distance between the ground truth and the estimation. Most objects will still have close surfaces after being rotated by $180°$, which explains the implied symmetry. As mentioned, for some use cases we want to measure the surface overlap/distance, but this is not ideal for the general use case of 6D pose estimation. In virtual reality, for example, drinking from a flipped coffee mug would not look right. The ADDS not only behaves different for every object, while having different upper bounds for every object, it is also much lower in general than the ADD metric. With upper bounds ranging from practically 0 cm to about 5 cm. This is important for threshold selection when calculating scores, since state-of-the-art evaluation uses the same threshold for ADD and ADDS, which can be seen in [36, 11, 29, 32, 23, 33, 34, 30].

In Figure 3.7 the ADD, ADDS, ACPD and MCPD metric are plotted under rotation for objects with a twofold symmetric axis. The objects were rotated from $0°$ to $360°$, along their twofold symmetric axis. The ADD metric ignores the symmetry of the objects entirely, treating them exactly the same as objects with one correct pose. The ADDS, ACPD and MCPD are all taking the symmetry of the objects into account. They all have two maxima at around $\pi/2$ radians and $3\pi/2$ radians, with the second minimum at around π radians. The ADDS metric again behaves different for different objects. For some objects, the error metric is close to zero, no matter the rotation angle. Which implies rotation invariant on

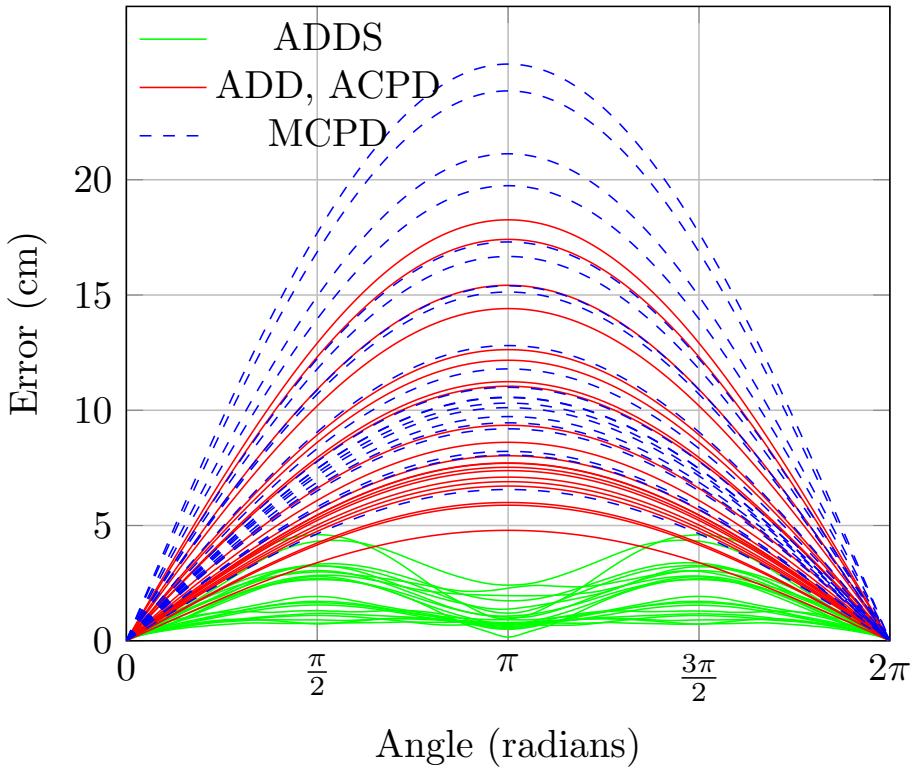

Figure 3.6: The ADD, ADDS, ACPD and MCPD metrics under rotation for objects with a single correct pose. Every object from the YCB-Video dataset was used. The dashed blue lines represent the MCPD, the red lines the ADD and ACPD and the green lines the ADDS. Each line represents the error for one object with the corresponding metric. On the X-axis the rotation angle is plotted and on the Y-axis the resulting error. It can be seen, that all metrics have different upper bounds depending on the object.

a symmetric axis. Also for one object the minimum at π is quite high at around 1.5 cm, when the maximum for this object is around 3 cm. Of course, since the points in the point cloud of the estimation do not align perfectly with points of the ground truth, a small deviation from zero can be expected, but these small deviations can add up to a substantial error. The spread between upper bounds of different objects is quite large in this case, ranging from practically 0 cm to about 3 cm.

The ACPD and MCPD both are behaving similarly. The ACPD takes the minimum of two ADD curves, where one is shifted by π radians. This results in a clean second minimum at π radians and maxima aligned with the symmetry of the object. There is, however, still a large amount of spread in the upper bounds for different objects. The maximum values for the ACPD are between about 4 cm to 10.2 cm. This also applies to the MCPD, which has maximum values ranging from about 5.1 cm to almost 15 cm. In Figure 3.8 the ADD, ADDS, ACPD and MCPD are plotted under rotation for objects with a rotation invariant axis. The YCB-Video dataset only contains one object that is classified as having a rotation invariant axis. However, since the metrics that actually take rotation invariance into account, do almost not depend on the object under consideration, the study of only one object is enough to understand how the metrics behave with regard to rotation invariance. The ADD metric ignores the rotational invariance entirely and treats every object as if it had only one correct pose. Since the ADD metric measures each individual point distance, from a single ground truth to a pose estimation, this is true for every object. The ADDS metric produces error values near zero for every rotation. The values are not exactly zero, since the ground truth and estimated point cloud do not align perfectly, but it is close enough for evaluation purposes. It can be assumed, that the point cloud of an object with a rotation invariant axis aligns close enough with the estimation, for any rotation on the rotationally invariant axis, which makes the result transferable to other objects. This is, however, under the assumption, that the point cloud is of sufficient quality. In the extreme case of only point, in a point cloud, the ADDS metric would behave similarly to the ADD metric. This means the ADDS requires a dense and evenly distributed point cloud. If a higher density point cloud was available, the ADDS metric would be even closer to zero, for every error angle. The ACPD and MCPD are both nearly exactly zero for every rotation angle. The deviation from zero is due to the numerical instability of floating point numbers. For all practical purposes, they can be considered to be zero.

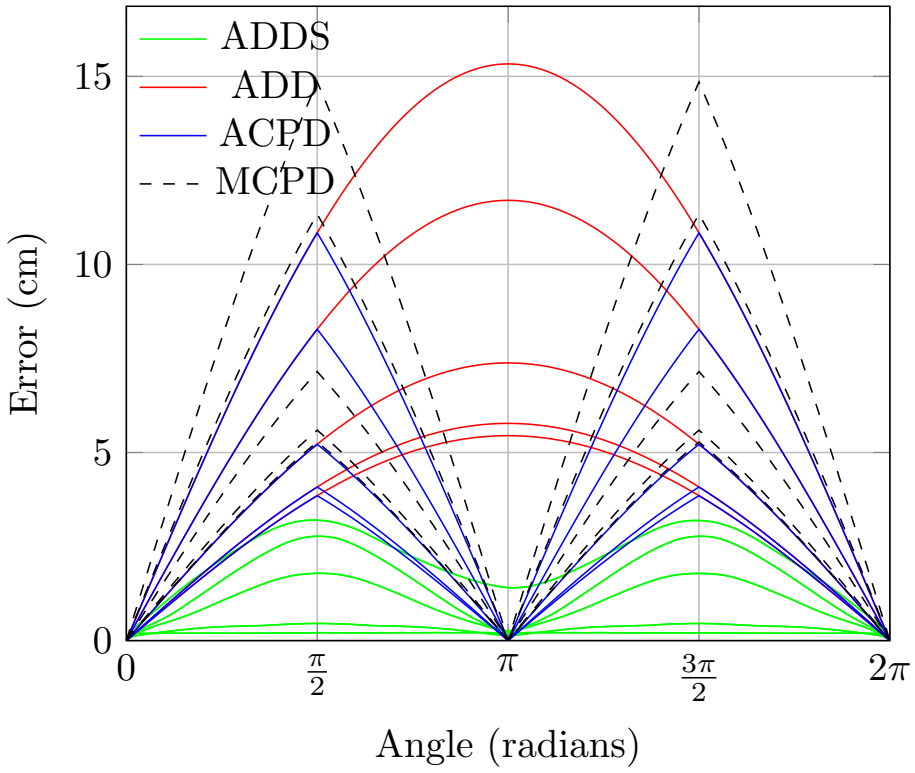

Figure 3.7: The ADD, ADDS, ACPD and MCPD metrics under rotation for every YCB-Video object with a twofold symmetric axis. The objects were rotated around their twofold symmetric axis. The rotation angle is plotted on the X-axis and the resulting error on the Y-axis. The ADD metric is represented by the red lines, the ADDS by the green lines, the ACPD by the blue lines and the MCPD by the dashed black lines. Each line represents the error for one object. It can be seen that the ADD metric is not able to capture symmetries and the ADD, ADDS, ACPD and MCPD all have different upper bounds for different objects.

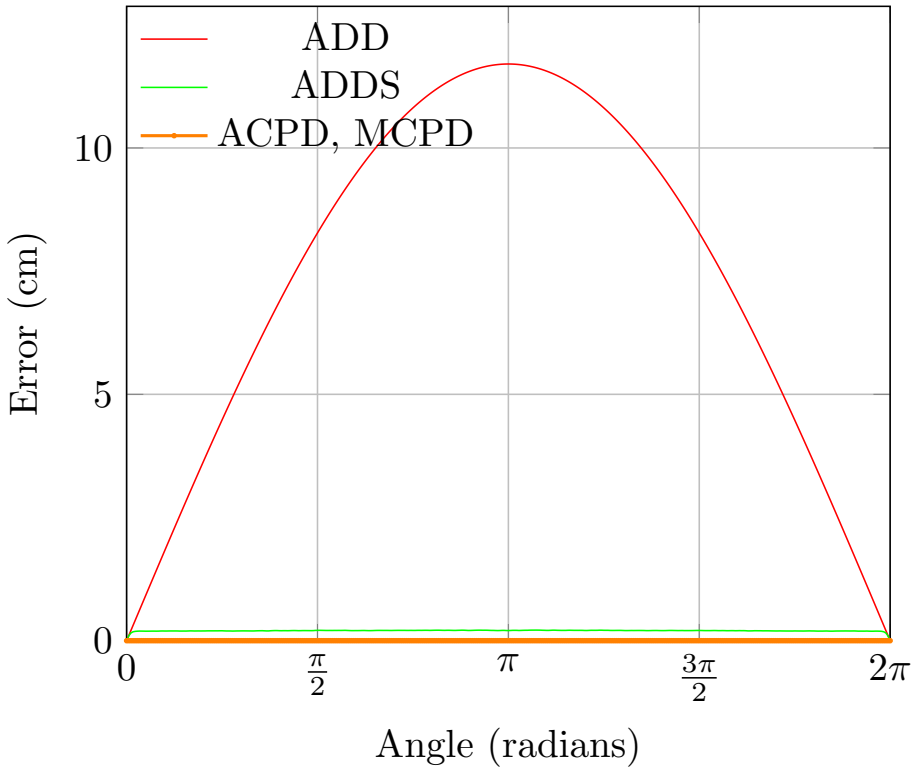

Figure 3.8: The ADD, ADDS, ACPD and MCPD metrics under rotation for an object with a rotation invariant axis, from the YCB-Video dataset. The object was rotated around its rotation invariant axis. The rotation angle is plotted on the X-axis and the resulting error on the Y-axis. The red line represents the ADD metric, the green line the ADDS metric and the orange line the ACPD and MCPD. It can be seen that the ADD metric ignores the rotation invariance entirely, while the ADDS, ACPD and MCPD produce error values close to zero for every rotation angle.

3.3.3 RESULTS UNDER POINT CLOUD CHANGES

The MRTE is not dependent on the specific point cloud used. The ADD, ADDS, ACPD and MCPD in contrast, are dependent on the point cloud. In Figure 3.9 the ADD, ADDS, ACPD and MCPD are plotted under size scaling of the point cloud are plotted. Because the MRTE is not dependent on the point cloud, no plot is provided, since MRTE error value would remain constant. For the plot, every object of the YCB-Video dataset was used with a pose estimation that is wrongly rotated by $90°$ on the X-axis and the point cloud scaled from 0 to 10 times its original size. Since the objects have only correct pose, the ADD and ACPD metric are equal. The ADD, ADDS, ACPD and MCPD are all increasing, when the size of the point cloud is increased. Since the ADD, ADDS, ACPD and MCPD are all measuring distances of points, this was expected. The same metric, ADD for example, is growing at different rates for every object, under point cloud size increase. This means, when increasing the size of every object in the dataset by the same factor, the error metric values do increase by different factors, which makes it difficult to set a threshold for score calculations for all objects. To counteract this, [13] proposed to use a threshold given as a proportion of object size. For example, ADD AUC with a threshold of 10% of object size. However, this approach does not deal with the fact that each object has different rate at which the ADD, ADDS, ACPD and MCPD increase under size changes. This issue can be especially problematic when a dataset contains objects of strongly varying sizes, for example, cars, lorries and bicycles, or tables and bottles. When using a large fixed threshold, the small objects will easily get accepted even when larger errors a present, when using a small threshold the large objects will not get accepted, even when the error is small.
Other changes to the point clouds should also be considered, like density, for example. A sparse point cloud might miss important features of an object, which for example makes it difficult to identify its rotation. An example of this is a bowling ball, a sparse point cloud might miss the three finger holes, which makes the bowling ball point cloud indistinguishable from a ball point cloud.

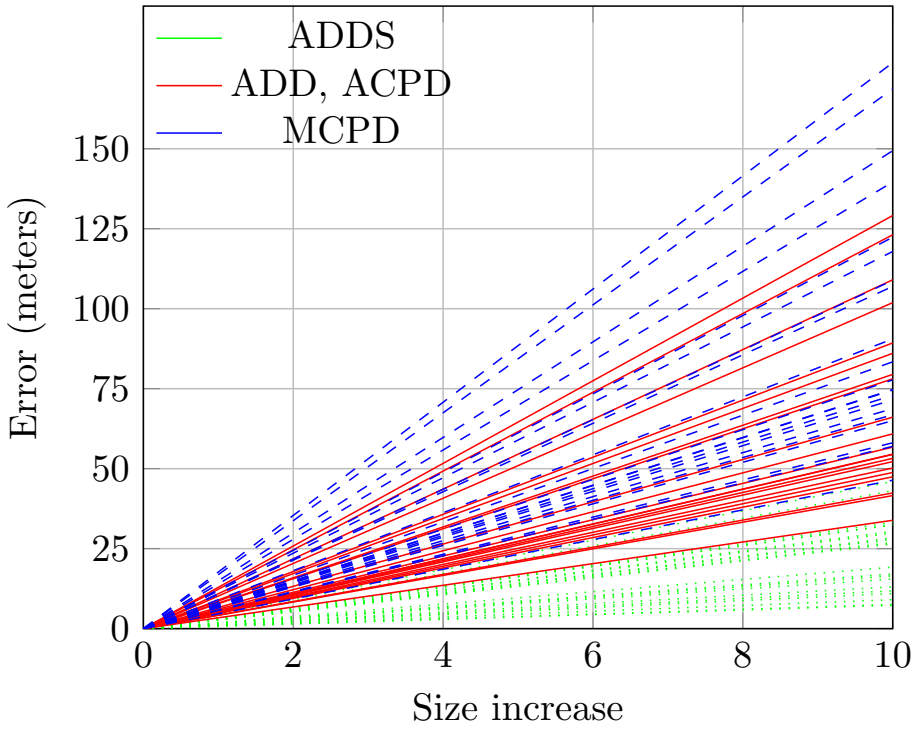

Figure 3.9: The ADD, ADDS, ACPD and MCPD metrics under point cloud size scaling. Every object in the YCB-Video dataset was scaled from 0 to 10 times its original size and has a pose error of 90° on the X-axis. The size increase is plotted on X-axis and the resulting error on the Y-axis. The ADD and ACPD metric represented by the red line, the ADDS by the dotted green line and the MCPD by the dashed blue line. Each line represents one object. It can be seen that ADD, ADDS, ACPD and MCPD all increase with point cloud size. The rate at which they increase is different for every object.

3.3.4 THOUGHTS ON THE AVERAGE DISTANCE OF MODLE POINTS METRIC

The main advantage of the e_{ADD} metric is its simplicity. It is also stable under translation. The main disadvantage is that the ADD metric can not be used on objects with more than one correct. It also is dependent on the propetries of thepoint cloud used, leading to diffrent upper bounds for every object.

3.3.5 THOUGHTS ON THE AVERAGE DISTANCE OF MODLE POINTS FOR SYMMETRIC OBJECTS METRIC

The ADDS metric addresses the main disadvantage of the ADD metric by being usable on every object, independent on the amount of correct poses. Even though it is calculated similarly to the ADD metric, it measures a completely different aspect: the surface distance between the estimated and ground truth point clouds. Meaning, it is not comparable to the ADD metric. This can be seen in the plots, the ADDS metric is not stable under translation, or rotation and dependent on the point cloud. Also, the values produced are much smaller than from ADD. Making the evaluation approach, as proposed in [36] and used by [11, 29, 32, 23, 33, 34, 30, 15, 25, 31, 6, 18, 35, 8], of using ADD for objects with a single correct pose and for objects with multiple correct poses ADDS, with the same AUC threshold, not comparable. Since a good threshold for ADD will lead to a significantly higher score using ADDS.
For use cases where surface overlap is more important than object rotation, the ADDS metric is a good choice. However, taking the ADDS as the default evaluation metric will lead to extremely high scores which are unintuitive, i.e. the ADDS metric produces low error values for poses that are visually bad. For example, when the rotation is wrong, shifting the pose estimation with the translation vector $\hat{\mathbf{t}}$ can lead to better results. This was done in Figure 3.10, where adding translation makes the object surface align more. Also, the pose estimation is rotated by 180° and translated by -1 cm on the Z-axis and the ADDS error is still lower than expected at 0.58 cm, for such a big error in the rotation.

3.3.6 THOUGHTS ON THE TRANSLATION ERROR METRIC

The translation error only measures translation, i.e. the rotation is ignored entirely when being used on its own. It is intuitive for quantifying the translation error in an estimated pose and does not rely on the point cloud used.

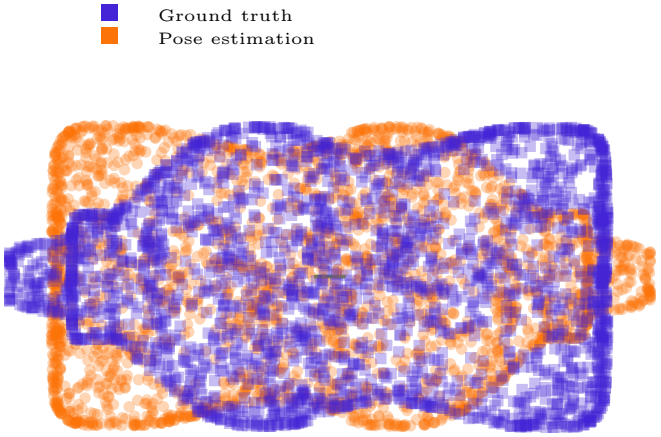

Figure 3.10: A visualization of a bad pose estimation (orange), for a mustard bottle, from the YCB-Video dataset. The ground truth (blue) was rotated by 180° on the Y-axis and translated by -1 cm on the Z-axis. The error under the ADDS metric would be 0.58 cm. This would result in a score of 94.2%, using the AUC with a starting threshold of 10 cm, which is the evaluation method used by state-of-the-art pose estimators. The proposed AIMRTES would provide a score of 47.8% for this pose estimation.

3.3.7 THOUGHTS ON THE ROTATION ERROR METRIC

The rotation error has two drawbacks. The first one being that translation is ignored entirely, and the second being that it can only be used on objects with a single correct pose. It does however measure the rotational error for objects with one correct pose perfectly.

3.3.8 THOUGHTS ON THE COMPLEMENT OVER UNION METRIC

When theCoU with bounding boxes is used, every object is treated like a box. Since a box has an axis with a twofold and another fourfold symmetric axis, an accurate representation of the rotation error in the pose can not be expected. Furthermore, some objects will have large areas in the box which are not part of the object, meaning that the bounding boxes might overlap, but the objects do not. An example is an L-shaped object, where the upper right corner of the bounding box will not contain any part of the object. Also, worse pose estimations can lead to a better metric value. An example is a stick, when no rotation is applied moving one stick a bit can make them no longer overlap. By rotating one stick, so that they are no longer parallel, they can overlap again resulting in a better metric, even tough the rotation estimation was deteriorated.

When using the object area instead of bounding boxes, the same example with sticks can be applied. When using object volume, the metric behaves similar to the ADDS metric, since when the area of the estimation and ground truth are overlapping there will also be some surface overlap. TheCoU has a maximum of 1, when the area of the objects or bounding boxes are no longer overlapping. For use cases where volume overlap is important, this metric is a good choice. However, using it as a default metric leads to unintuitive metric score values, since a low metric does not necessarily imply the estimated rotation is close to the ground truth rotation.

3.3.9 THOUGHTS ON THE AVERAGE CORRESPONDING POINT DISTANCE METRIC

The ACPD tries to extend the ADD metric to address its main drawback of only being applicable to objects with one correct pose. The other drawbacks of being heavily dependent on the point cloud used and producing different upper bounds for every object remain. Additional, when it was introduced in [14] no method for finding the additional correct poses was described. The additional correct poses can be found as described in Section 3.2 i.e. the same method of adding rotations to the ground truth, as for the MRE, can be used, which makes the metric usable with current datasets like the YCB-Video dataset.

3.3.10 THOUGHTS ON THE MAXIMUM CORRESPONDING POINT DISTANCE METRIC

The MCPD is essentially a stricter version of the ACPD. Since it only uses one point pair of the estimated and ground truth point cloud, it can be argued that it is less dependent on the density of the point cloud. In practise, however, high density point clouds are almost always available.

3.3.11 THOUGHTS ON THE VISIBLE SURFACE DESCRAPANCY METRIC

The VSD is similar to the ADDS metric, it also measures surface distance. The main difference being that only the visible surface of the ground truth and estimation is used in the metric, which leads to an even more lenient metric. An argument can be made, that only using the visible surface is a fairer way of evaluation, since no information of the not visible parts of the objects is available. In [14] the example of a coffee mug is used, without seeing the handle it is impossible to reliably find the real rotation of the mug. The VSD is however not perfect in this regard. It does not take texture information into account, which means that even when the correct rotation could be found from colour information, the VSD will ignore this. The coffee mug form [14] was

monochrome. If the mug had a graphic printed on it, the single correct pose can be identified from almost any view angle.
Also, for occluded objects the visible surface becomes rather small, making it easier to satisfy the VSD metric. The VSD is also more complex than the other metrics, needing projections from 3D space to 2D space.

3.3.12 THOUGHTS ON THE ACCURACY SCORE

The accuracy score is easy to understand, but for the purpose of 6D pose estimation, it has many flaws. Since it only uses a single fixed threshold, it converts the problem to a binary classification problem, where a pose is either correct or incorrect. If the threshold τ is for example 5 cm using the ADDS metric, a pose estimator that is always of off by 4.9 cm in the estimation is considered perfect. Which makes comparisons only possible if all errors are around the given threshold. False detections are ignored entirely. It could, however, be extended to address false detections by also iterating over detected objects in addition to the ground truth, like it is proposed for the AIMRTES.

3.3.13 THOUGHTS ON THE MEAN RECALL SCORE

The MR, as it is defined in [14] and can be viewed as the average accuracy over multiple thresholds, which is an improvement since better pose estimations can now score higher, when the error metric is lower than multiple thresholds. The problem of false detections not being addressed still remains, but can be addressed like it is done with the AIMRTES.

3.3.14 THOUGHTS ON THE AREA UNDER CURVE SCORE

The AUC is essentially the MR with an infinite amount of thresholds $\mathcal{T}$ from 0 to a threshold γ. Which means that every improvement in the estimated pose now counts towards the score, instead of predefined steps, but every pose worse than γ is treated the same. With its standard definition, it also ignores false detections entirely, but this can again be addressed by iterating over the detected objects alongside the ground truth.

3.3.15 THOUGHTS ON THE SYMMETRIC ROTATION ERROR METRIC

The symmetric rotation error is an extension of the rotation error, addressing the drawback of not being able to be used on symmetric objects.

3.3.16 THOUGHTS ON THE ROTATION INVARIANT ROTATION ERROR METRIC

The rotation invariant rotation error extends the rotation error to make it applicable to be used on objects with a rotation invariant axis.

3.3.17 THOUGHTS ON THE MUTLI ROTATION ERROR METRIC

The multi rotation error combines the symmetric rotation error and the rotation invariant rotation error, to create a rotation error metric that can be used on any object. It does not depend on the point cloud used and has predictable upper bounds. Its main drawback is its inablity to address translation errors.

3.3.18 THOUGHTS ON THE MULTI ROTATION TRANSLATION ERROR METRIC

The MRTE combines the multi rotation error with the translation error. This deals with the two main drawbacks of the individual metrics and is able to address both rotation and translation errors. Its advantages are, that it can be used on any object, does not depend on the point cloud used and has predictable upper bounds. Through scaling, the MRTE is applicable to a wide verity of use cases.

3.3.19 THOUGHTS ON THE AVERAGE INVERSE MULTI ROTATION TRANSLATION ERROR SCORE

The AIMRTES was designed to address the issue of false detections alongside the pose estimation errors. Instead of just taking the objects of the ground truth dataset into account, it also iterates over all detected objects. Since no threshold is defined, there is no cut off for any pose estimator, meaning a higher scoring pose estimator always performance better.

3.3.20 SUMMARY

The ADD and ADDS in combination with the AUC is the most commonly used method for evaluation. As discussed, this method has significant disadvantages, especially when the same AUC threshold for both ADD and ADDS is used.
The ACPD and MCPD try to address some of the issues, but not all. Also, since no method for finding other correct poses was provided, these metrics never caught on in practical applications.
The MRTE alongside with AIMRTES, is the only evaluation method which fulfils all criteria of a good evaluation method. It addresses rotation errors, translation errors and false detections equally and predictably. Can be used on any object and treats objects independent of the point cloud.

4 Experiments

This chapter shows how the proposed evaluation method performs for the use case of evaluating 6D pose estimators under disturbances. The task of examining pose estimators under sensor disturbances is crucial for real-world applications, as it provides insights into the robustness and reliability of these systems in practical scenarios. Various factors such as noise, occlusions, and dynamic environments, can significantly impact the accuracy of pose estimation, making it essential to assess how well these systems can maintain performance under such conditions. By systematically introducing and analysing disturbances in sensor data, we can better understand the limitations and strengths of pose estimators.

4.1 SETUP

In this section, the general setup for the following experiments is explained. For the case study of how 6D pose estimators behave under disturbances, FFB6D is used together with the YCB-Video dataset. Both are described in detail in Chapter 2. FFB6D was chosen because it uses a bidirectional fusion approach in the convolution layers, how this fusion approach deals with errors in one data source has yet to be addressed. FFB6D also has an impressive ADDS AUC of 96.1% on the YCB-Video benchmark.
The YCB-Video dataset was chosen due to its size and high quality ground truth poses for common household objects.
The approach of [3] was followed when conduction the experiments. This means that the YCB-Video dataset will be artificially enhanced to also contain disturbances. FFB6D will be benchmarked on each disturbance individually, under different error intensities.
For each intensity the ADD AUC, ADDS AUC, AIMRTES without false detections (w. f.d.) , AIMRTES with false detections, average scaled (avg. s.) multi rotation error with standard deviation, average scaled translation error with standard deviation and percentage of false detections are provided. The AIMRTES without false detections only takes the error values of poses inside the evaluation dataset

into account. The AIMRTES with false detections also takes the wrongly detected object poses into account, like it is defined in Section 3.2.5. The ADD and ADDS AUC are provided since they are the current state-of-the-art evaluation method, with which the proposed evaluation (AIMRTES) will be compared to. Additionally, the rotation error, translation error and amount of detected objects are provided, to better understand what kind of error a disturbance introduces. If the rotation error, translation error, or amount of false detections increases, the scores should decrease.
The AUC threshold is set to 10 cm as proposed in [36] and used by [11, 29, 32, 23, 33, 34, 30] because of this the scaling $g(x)$ in the MRTE (used by the AIMRTES) will scale the translation error by dividing it by 10 cm and cutting off values higher than 10 cm, i.e. $\beta = 10$ cm. The MRE, in the MRTE is scaled so that it is between zero and one. This means that the AIMRTES values 1 cm of translation as much as 36° of rotation. These scaling are also used on the average MRE and translation error, of course missed objects and false detections are excluded for the average since they have an error of ∞, also the translation error is not cut off after 10 cm for the average calculation. The YCB-Video benchmark contains 14000 ground truth poses, meaning that if 7000 false detections were made the false detection rate is 50%.

4.1.1 SENSOR DISTURBANCES

In this section, the errors featured in the experiments are introduced. As described in Section 2.1.3 the Asus Xtion used by the YCB-Video dataset uses an infrared structured light sensor for its depth images. This sensor is highly prone to disturbances, since any other infrared light source can interfere with the distance calculation. The surface texture of objects can also lead to errors in the distance calculation, depending on how the light is reflected. This can lead to wrong distance reading of some pixels.
For both the RBG and infrared camera, the lens can be blocked, resulting in missing spots in the RBG or depth image. Also, motion blur can accrue in both sensors when the sensor setup (Asus Xtion) is moved. Both sensors can also experience hardware failure, which can results in dead pixels.[1]
In the following experiments, the following disturbances are considered:

Missing spots in the depth image This can happen when something is blocking the view of the infrared camera, for example, dirt or debris in on camera lens.

[1] An implementation of all disturbances can be found under `https://github.com/D-Doge/ffb6_disturbances`.

Missing spots in the RGB image This can happen when something is blocking the view of the RGB camera, for example, dirt or debris on the camera lens.

Noise in the depth image Other infrared emitters, like the sun, can cause noise in the depth image.

Noise in the RGB image Noise in the RGB image can be caused by low light conditions, resulting in graininess, or by electrical interference affecting the camera sensor, leading to unwanted artefacts or colour distortions in the image.

Motion blur in the depth image Motion blur can occur when the camera is moved during the recording of a frame.

Motion blur in the RGB image As for the depth image motion blur can also occur in the RGB image.

4.2 BASELINE

Before examining how FFB6D performance under disturbances, the baseline performance on the YCB-Video benchmark without disturbances is shown in Table 4.1. Even without disturbances, the amount of false detections is already high. The YCB-Video benchmark contains 14,000 ground truth poses. FFB6D detects 21672 objects, resulting in a false detections rate of 54.8%. The high false detections rate is probably due to the fact, that the current evaluation metrics ignore the aspect of false detections entirely. The ADD AUC and ADDS AUC scores are both quite good, at 92.4% and 96.1% respectively. The AIMRTES without false detections is also good at 91.2%, since for these experiments a similar scaling as for the ADD and ADDS AUC was chosen. In contrast, the AIMRTES takes false detections into account and is much lower at 58.6%.
The average translation error is extremely low at 4 mm. Since FFB6D was trained with the ADD metric as the loss function, this is explainable, since the ADD metric punishes translation errors more than rotation errors, which is shown in Chapter 3. The average MRE is about 0.25, which corresponds to 15.5°.

ADD AUC	ADDS AUC	AIMRTES	AIMRTES w. f.d.	Avg. e_{R}	Avg. e_t	Percentage of f.d.
92.4%	96.1%	58.6%	91.2%	0.27	4 mm	54.8%

Table 4.1: This Table shows the results of FFB6D on the normal YCB-Video benchmark. The ADD ACU, ADDS AUC and AIMRTES without false detections are all good at over 90%. The standard AIMRTES is much lower at 58.6%, because the false detection rate is high at over 50%. The average translation error is excellent at only 4 mm, while the MRE is just a bit worse at 0.27.

Figure 4.1: An example of a depth image with the disturbance of missing circles applied at an intensity of one.

4.3 MISSING SPOTS IN THE DEPTH IMAGE

To simulate missing spots in the depth image, circles at random spots of the depth image were set to zero. The circle have a radius from 50 to 100 pixels, chosen at random. The intensity is given by how many circles are cut out from the image. An example of an augmented depth image can be seen in Figure 4.1. The results be seen in Figure 4.2. The AIMRTES without false detections (w. f.d.) produces similar results to the state-of-the-art ADD and ADDS AUC, being slightly more strict. Since a similar scaling for the AIMRTES was chosen to make it comparable to the ADD and ADDS AUC, this was expected. As the intensity of the disturbance increases, the average MRE and translation error increase as well. The percentage of false detections does not change much. The AIMRTES with false detections is much lower than the ADD and ADDS AUC, but decreases slower. It decreases slower because of the high amount of false detections, which do not change much under higher intensity and are adding more terms to the sum of the AIMRTES, making is less sensitive to other rotation and translation. To counteract this, the rotation and translation error in the AIMRTES can be weighted higher.
When examining how missing spots in the depth image affect FFB6D, it can be seen that the average scaled translation error and the average scaled MRE both increase at about the same rate. The standard deviation of the translation error is however growing faster, meaning more outliers are present. Since the scores are still high, this suggests that the translation of a pose is either estimated close to the ground truth or completely wrong. The same is true for the MRE, to a lesser degree.

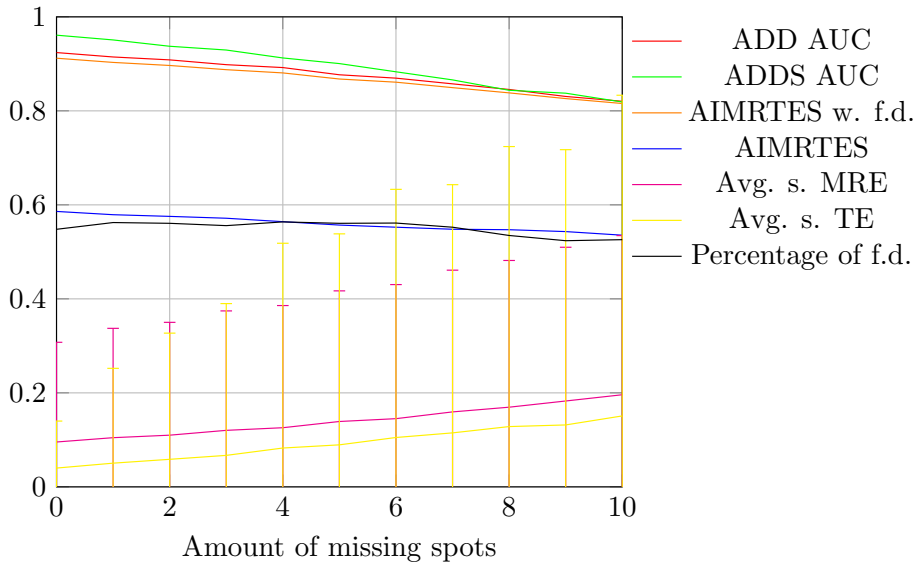

Figure 4.2: The plot shows the ADD AUC (red), ADDS AUC (green), AIMRTES without false detections (w. f.d.) (orange), AIMRTES with false detections (blue), average scaled (avg. s.) multi rotation error (magenta) with standard deviation (represented by the error bars), average scaled translation error (yellow) with standard deviation (represented by the error bars) and percentage of false detections (black). The intensity (how many spots are missing in the depth image) is plotted on the X-axis, with the corresponding value on the Y-axis. The ADD AUC, ADDS AUC and AIMRTES w. f.d. all behave similar, while standard AIMRTES is lower due to the high false detection rate. As the intensity of the disturbance is increasing, the error metrics are increasing slightly. It follows that the scores are decreasing slightly. Over all, FFB6D is not affect too harshly by the missing spots in the depth image.

4.4 MISSING SPOTS IN THE RGB IMAGE

The same method of setting random circles, with a radius from 50 to 100 pixel to zero, from the depth image is also used on the RGB image. An example is shown in Figure 4.3.
The results are shown in Figure 4.4. Compared to the missing spots in the depth image, the missing spots in the RGB image cause a stepper performance drop, in all four scores. This suggests, that the RGB information is more important to FFB6D than the depth information. The AIMRTES w. f.d. again performance similar to the ADD and ADDS AUC, which all decline under heavier disturbances. The AIMRTES without false detections also declines faster, for the disturbance on the RGB image.
The percentage of false detections is decreasing, since it makes little sense that FFB6D improves under disturbances, the RGB data is probably used to detected objects. With more parts of the picture missing, some objects might be cut out entirely, making them only detectable from the depth data. Because of the dropping false detection rate it can also be assumed that the true detection rate is dropping. The average scaled translation error and its standard deviation behave about the same, as for the missing circles in the depth image. The average scaled MRE and its standard deviation is higher, than for the missing circles in the depth image. This suggests that the RGB data is more important for rotation and about of the same importance for the translation. As both scaled errors get set to one, if the object is not found, it can not be due to decreasing detections rate.

Figure 4.3: An example of an RGB image with the disturbance of missing circles applied at an intensity of two.

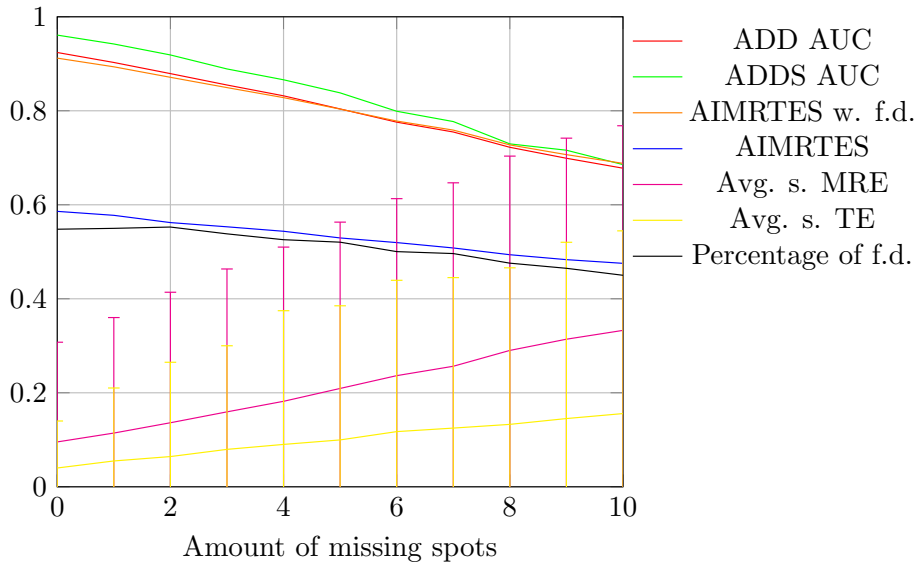

Figure 4.4: The plot shows the ADD AUC (red), ADDS AUC (green), AIMRTES without false detections (w. f.d.) (orange), AIMRTES with false detections (blue), average scaled (avg. s.) multi rotation error (magenta) with standard deviation (represented by the error bars), average scaled translation error (yellow) with standard deviation (represented by the error bars) and percentage of false detections (black). The intensity (how many spots are missing in the RGB image) is plotted on the X-axis, with the corresponding value on the Y-axis. Compared to the missing spots in the depth image, the average scaled MRE is increasing faster, which results in a larger drop in the scores. The false detection rate is also decreasing, which suggest that the RGB image play a critical role in detecting object. The standard deviation of the average scaled translation is lower for the disturbance in the RGB image, this suggests that the RGB image is not as important for translation.

4.5 NOISE IN THE DEPTH IMAGE

To simulate noise in the depth image a random value drawn from a Gaussian distribution is added to every pixel in the depth image. The intensity is given by the standard deviation of the Gaussian distribution. A value of one is equal to 100 μm, i.e. adding a value of ten, to a pixel in the depth image corresponds to adding 1 mm to the depth reading. This is why also some high values, up to 10,000 (1 m) are included. An example of a noisy depth image is shown in Figure 4.5.

The results can be seen in Figure 4.6. The average scaled translation error exceeds its limit of 10 cm after an intensity of 1000 by such a huge margin, that even the standard deviation is above 10 cm. This makes sense, since a pixel value of 1000 corresponds to 10 cm. The average scaled MRE is lower, which again suggest, that the depth data is mainly used for translation. The false detection rate is also increasing rather quickly after an intensity of 500. Which means the depth data also plays a role in object detection.

An intensity lower than 10 seems to have no effect. The data used to train FFB6D comes from a real Asus Xtion, where this level of noise might be present in the sensor, even under ideal conditions.

The ADD AUC is zero as soon as the translation error is above 10 cm. The ADDS AUC is zero after a little higher intensity, since it is not as sensitive to translation, which was shown in Chapter 3. After they reach zero it not possible to see if the performance of FFB6D gets better or worse, when only relying on the ADD and ADDS AUC. Both AIMRTES based scores are still decreasing, even after the translation error is maxed out, because the MRE is still getting worse. Which means it is still possible to see the performance decreasing, even when one error aspect is maxed out.

It has to be noted, that AIMRTES also awards true detections, since even objects with a bad pose estimation will add something to sum. In this case, a true detection will add $\frac{1}{3}$ to the sum. From this it can also be seen, that FFB6D does not detect every object in the benchmark dataset, because otherwise the AIMRTES without false detections could not be lower than $\frac{1}{3}$.

Figure 4.5: An example of a depth image with the disturbance of added Gaussian noise applied at an intensity of 10000.

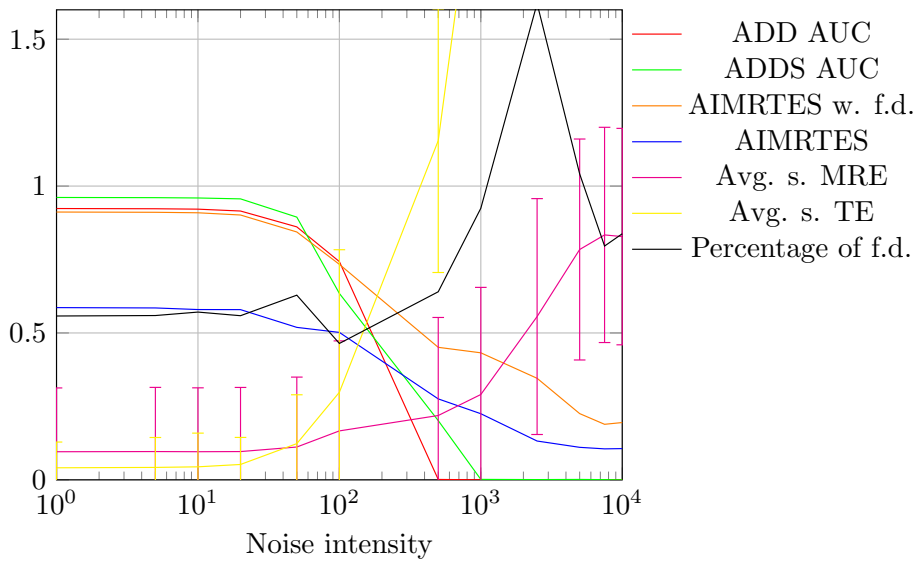

Figure 4.6: The plot shows the ADD AUC (red), ADDS AUC (green), AIMRTES without false detections (w. f.d.) (orange), AIMRTES with false detections (blue), average scaled (avg. s.) multi rotation error (magenta) with standard deviation (represented by the error bars), average scaled translation error (yellow) with standard deviation (represented by the error bars) and percentage of false detections (black). The intensity (standard deviation of the Gaussian noise added in the depth image) is plotted on the X-axis, with the corresponding value on the Y-axis. Low intensities up to 15 are not affecting FFB6D much. After an intensity of 100 the average scaled translation error is increasing fast, which results in the ADD and ADDS AUC dropping to zero after an intensity of 1000 is reached. This again solidify the notion that the depth image is important for translation. The AIMRTES based scores are still awarding the rotational aspect of the pose and the object detecting capabilities and are still decreasing as the intensity is increasing. The average scale MRE is increasing much slower than the average scaled translation error, which suggest that the depth information is not as important for rotation. The false detection rate is also increasing quickly after an intensity of 1000, but is also decreasing again, which suggest that the depth image is also involved in the object detection aspect of FFB6D.

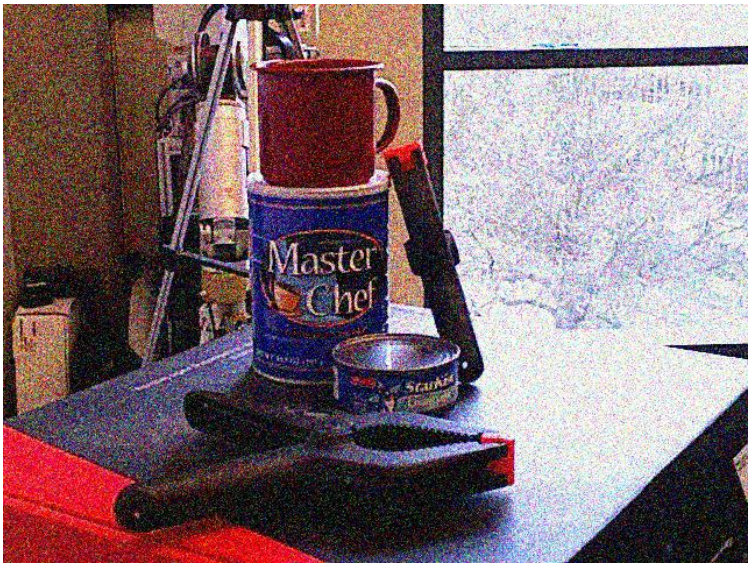

Figure 4.7: An example of an RGB image with the disturbance of added Gaussian noise applied at an intensity of 10.

4.6 NOISE IN THE RGB IMAGE

To simulate noise in the RGB image a random value drawn from a Gaussian distribution is added to every channel of every pixel. The intensity is given by the standard deviation of the Gaussian distribution. The image is interpreted as an eight bit three channel image, i.e. every channel contains values ranging from 0 to 255. An example of a noisy RGB image is shown in Figure 4.7.
In Figure 4.8 the results are shown. The false detection rate is increasing much faster than the average scaled rotation or translation error. The AIMRTES is the only scores that reacts to the increase in false detections, which is easy to see when look at an intensity range from about 0 to 80. The false detection rate is sensitive to noise in the RGB image, much more than for noise in the depth image, which again suggest that the RGB image is important for object detection. The noise intensity of the RGB can not be compared to the noise intensity of the depth image, since the values are encoding different information (colour and distance). After an intensity of around 100 the average scaled rotation and translation errors are also increasing fast, which causes the ADD AUC, ADDS AUC and AIMRTES without false detections to drop. As the false detection rate increases, the AIMRTES becomes less sensitive to rotational and translational errors, because the rotational and translational errors make up a smaller fraction of the sum over which the AIMRTES is averaged.

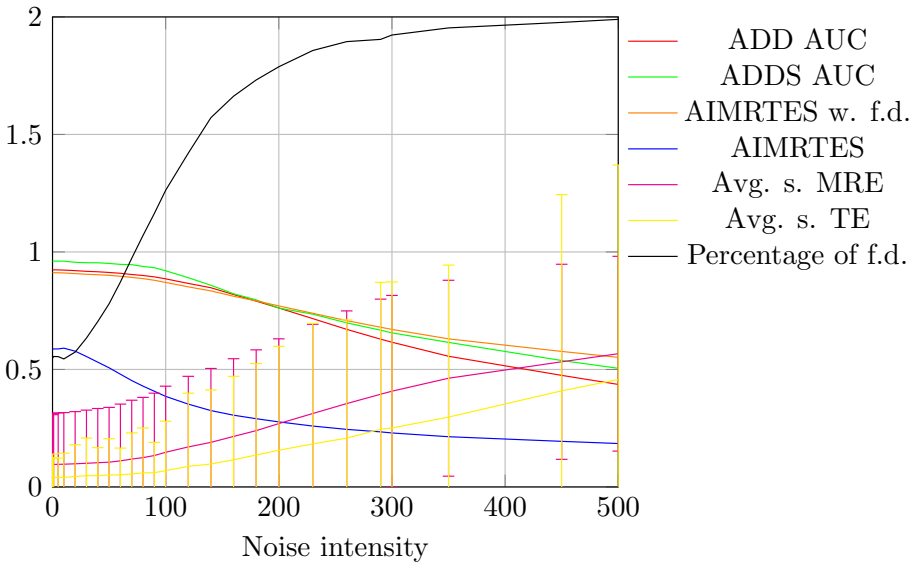

Figure 4.8: The plot shows the ADD AUC (red), ADDS AUC (green), AIMRTES without false detections (w. f.d.) (orange), AIMRTES with false detections (blue), average scaled (avg. s.) multi rotation error (magenta) with standard deviation (represented by the error bars), average scaled translation error (yellow) with standard deviation (represented by the error bars) and percentage of false detections (black). The intensity (standard deviation of the Gaussian noise added in the RGB image) is plotted on the X-axis, with the corresponding value on the Y-axis. The false detections rate is affected the most by the disturbance, reaching over 100% at an intensity of 80, which causes the AIMRTES to decline faster than the other scores, which is the only score to take false detection into account. The fast increase in false detections suggest that the RGB image plays a major role for the object detection.The average scaled rotation error is also increasing faster than the average scaled translation error.

Figure 4.9: An example of an RGB image with the disturbance of added Gaussian noise applied at an intensity of 10.

4.7 MOTION BLUR IN THE DEPTH IMAGE

To simulate motion blur in the depth image, a linear motion blur is applied using a convolutional kernel that smears pixel values along a randomly chosen direction. The length of the blur is the intensity and is given in pixels. The direction of the motion blur is determined by a randomly generated angle. An example of this disturbance can be seen in Figure 4.9.
The average scaled rotation error is only increasing very slowly, which suggest that the depth image is not important for finding rotations. The average scaled translation is increasing faster again, which solidifies that the depth image is used for finding translations. The false detection rate is also increasing fast. The ADDS AUC is much lower than the ADD AUC, even though ADDS is a more lenient metric than ADD, this suggest, that some objects with multiple correct poses get affected more by the motion blur disturbance. The AIMRTES without false detection is decreasing much more slowly than the ADD and ADDS AUC. This is probably due to the fact, that the AIMRTES w. f.d. is still awarding the low rotation error, where the ADD and ADDS AUC classify the pose as incorrect, due to the high translation error.

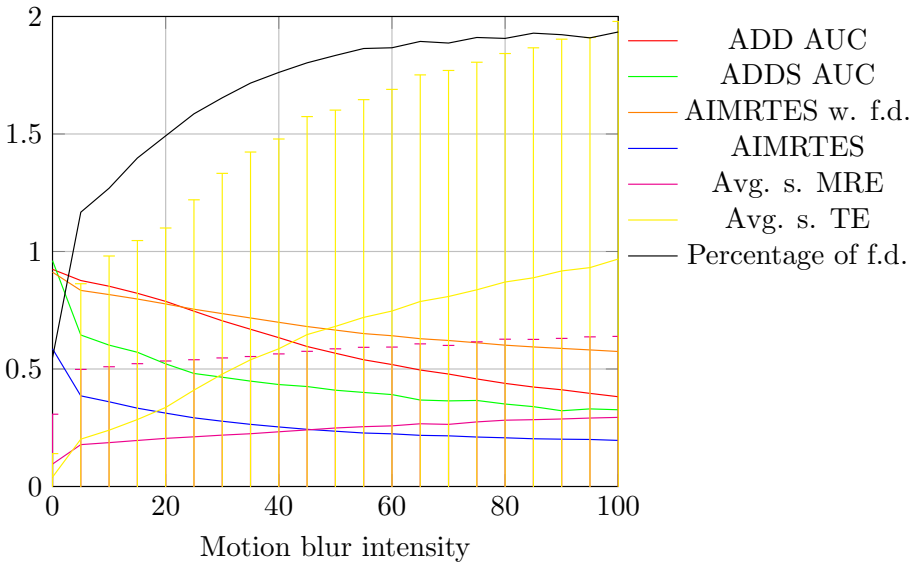

Figure 4.10: The plot shows the ADD AUC (red), ADDS AUC (green), AIMRTES without false detections (w. f.d.) (orange), AIMRTES with false detections (blue), average scaled (avg. s.) multi rotation error (magenta) with standard deviation (represented by the error bars), average scaled translation error (yellow) with standard deviation (represented by the error bars) and percentage of false detections (black). The intensity (length of the motion blur in the depth image) is plotted on the X-axis, with the corresponding value on the Y-axis. The false detection rate and the averaged scaled translation error are increasing fast. The averaged scaled rotation error is increasing very slowly. Which again suggest that the depth image is mainly used for detection and translation. The ADDS AUC is decreasing faster than the ADD AUC, which suggest that some objects with multiple correct poses are affected more by the motion blur, since the ADDS is a more lenient metric than ADD. Because the average scaled translation error is much higher than the average scaled rotation error, the AIMRTES is higher than both the ADD and ADDS AUC. This is probably due to the fact the AIMRTES w. f.d. still awards the low rotation error.

Figure 4.11: An example of an RGB image with the disturbance of added Gaussian noise applied at an intensity of 10.

4.8 MOTION BLUR IN THE RGB IMAGE

The motion blur for the RGB image was simulated with a convolutional kernel, as for the depth image. The convolutional kernel was applied to every channel of the RGB image. This disturbance was included in the training of FFB6D with an intensity of up to 15. An example of the applied motion blur can be seen in Figure 4.11.
The results are plotted in Figure 4.12. The false detection rate is much better than for the other disturbances, especially in the intensity range included in the training. Both the average scaled rotation and translation error are only increasing very slowly for intensities below 15. This means that including the disturbance during training seems to increase performance, however to gain more confidence in that regard this experiment should also be done on a version of FFB6D which was not trained with this disturbance. The average scaled rotation error is again increasing fast, which solidifies that the rotation is mainly recovered from the RGB image. The scores behave as expected, with AIMRTES w. f.d. being again a bit higher than the ADD AUC and ADD AUC, due to the discrepancy in rotational and translational errors.

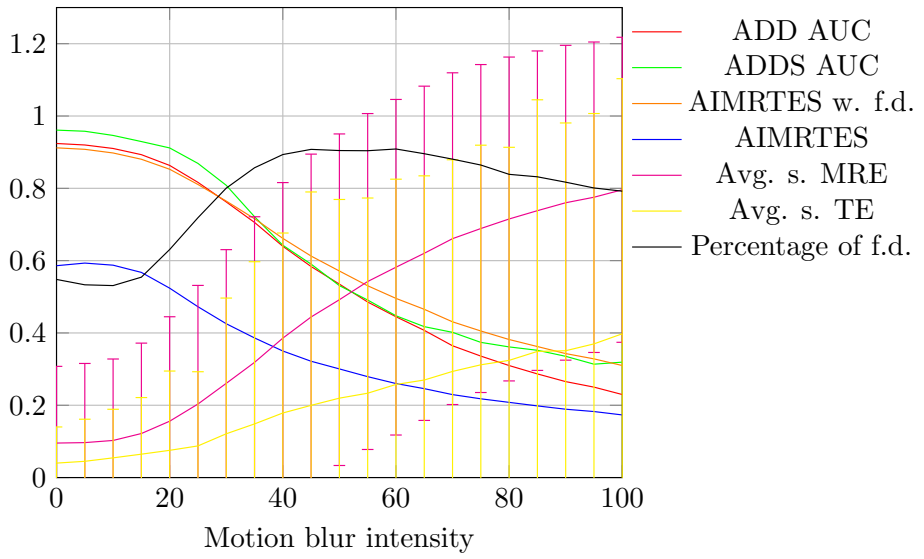

Figure 4.12: The plot shows the ADD AUC (red), ADDS AUC (green), AIMRTES without false detections (w. f.d.) (orange), AIMRTES with false detections (blue), average scaled (avg. s.) multi rotation error (magenta) with standard deviation (represented by the error bars), average scaled translation error (yellow) with standard deviation (represented by the error bars) and percentage of false detections (black). The intensity (length of the motion blur in the RGB image) is plotted on the X-axis, with the corresponding value on the Y-axis. The false detection rate is lower than expected, it even is decreasing at the start, when the intensity is blow 15. The average scaled rotation error is again increasing faster than the translation error. The ADD AUC, ADDS AUC, AIMRTES w. f.d. and AIMRTES are all decreasing as expected. It can be noted that FFB6D was trained with this disturbance, with an intensity of up to 15, which seems to have improved the performance in that intensity range.

4.9 SUMMARY

The AIMRTES is better suited than ADD or ADDS AUC for examining pose estimators under disturbances, because it does not reach zero as quickly, as can be seen in Section 4.5. This means, that the AIMRTES has a larger range of intensities, for which a difference in performance can be seen. Additionally, the AIMRTES is the only score that can take false detections into account, while performing similar to the ADD and ADDS AUC, when ignoring false detections. It is still recommanded to also track the average MRE, average translation error and false detections rate and maybe even the true detection rate, so it is clear which aspects of the pose estimator are affected by the disturbance.
By isolating disturbances in different data channels, it is possible to see how the pose estimator uses the provided information. In the case of FFB6D, it is apparent that the depth data is mainly used for translation and detection. The RGB data has more influence on the rotation than the translation and is also used for detections.

5 Conclusion

The newly developed evaluation score AIMRTES is able to address every aspect of 6D pose estimation. It can be applied to every kind of object and is unbiased to the different kind of objects, while considering every kind of error. The MRTE provides easy interpretable results, growing linear for both rotational and translation errors. Additionally, the scaling in the MRTE can be used to adapt the metric for different use cases, where rotation, translation and false detections are of different importance. The AIMRTES also does not treat pose estimation as a binary problem, i.e. it can distinguish the performance of pose estimators more precisely, than current state-of-the-art methods.
To analyse pose estimators under disturbances, the approach of using the AIMRTES alongside the average scaled rotation and translation error, while tracking the amount of false and true detections, is able to clearly isolate the different kind of errors, while providing a good indication of overall performance, in the form the AIMRTES.
For the example of FFB6D, it can be clearly seen that the depth image is highly important for the translation. This information can not be easily recovered from the state-of-the-art evaluation methods. Overall FFB6D deals quite well with minor disturbances.

5.1 FUTURE WORK

A logical next step would be to survey current state-of-the-art pose estimators using the AIMRTES, as current state-of-the-art evaluation does not take false detections into account, the results might greatly differ from the ADDS AUC. Also, using a stricter scaling than 10 cm should also be considered, since current scores are quite high, sometimes even as high as 99.7% ADD accuracy, as reported in [11].
Using the complement of the AIMRTES as the loss function in training for deep learning networks might improve results. Especially in regard to false detections, since currently often the ADD metric is often used as a loss function. The MRTE metric used by the AIMRTES also grows linearly for both rotation and translation and is unbiased for different objects. The gradient of the MRTE

is however not differentiable at its extrema, but this is a well known problem in machine learning. This problem is easily addressed by setting the gradient to zero at these extrema, which is also done for loss functions like the mean absolute error as can be seen in [5, p. 46].
With the newly introduced approach to analyse pose estimators under disturbances, it would also be interesting to try to improve FFB6D in terms of its robustness to disturbances. This could be archived by adding the disturbances that effect FFB6D to the training data, following the approach of [3]. With the finer amount of information on the different kinds of error, it is possible to see if other errors degenerate as overall performance increases. For example, if the training puts more focus on depth disturbances improving translation errors, the overall performance might increase while the rotational errors worsen. It might also be advantageous to include the true positive detection rate for the analysis, especially for pose estimators that only output a detection under higher confidence.

Bibliography

[1] R. Arrais. 2015. *Automatic Inconsistency detection in a Logistic World Model (European Project STAMINA)*. PhD thesis. (July 2015). DOI: 10.13140/RG.2.1.3341.2086 (cit. on p. 4).

[2] C. Bartalucci, R. Furferi, L. Governi and Y. Volpe. 2018. A survey of methods for symmetry detection on 3d high point density models in biomedicine. *Symmetry*, 10, 7, 263 (cit. on p. 8).

[3] F. Berens, Y. Koschinski, M. Badami, M. Geimer, S. Elser and M. Reischl. 2024. Adaptive Training for Robust Object Detection in Autonomous Driving Environments. *IEEE Transactions on Intelligent Vehicles*, PP, (Jan. 2024), 1–15. DOI: 10.1109/TIV.2024.3439001 (cit. on pp. 39, 60).

[4] P. Besl and N. D. McKay. 1992. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14, 2, 239–256. DOI: 10.1109/34.121791 (cit. on p. 5).

[5] C. M. Bishop. [2006]. *Pattern recognition and machine learning*. Textbook for graduates.;Includes bibliographical references (pages 711-728) and index. New York : Springer, [2006] ©2006 (cit. on p. 60).

[6] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton and C. Rother. 2014. Learning 6d object pose estimation using 3d object coordinates. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*. Springer, 536–551 (cit. on p. 33).

[7] E. Corona, K. Kundu and S. Fidler. 2018. Pose Estimation for Objects with Rotational Symmetry. *CoRR*, abs/1810.05780. arXiv: 1810.05780 (cit. on p. 19).

[8] B. Drost, M. Ulrich, N. Navab and S. Ilic. 2010. Model globally, match locally: Efficient and robust 3D object recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*. Ieee, 998–1005 (cit. on p. 33).

[9] R. Hartley and A. Zisserman. 2003. *Multiple View Geometry in Computer Vision*. (2nd ed.). Cambridge University Press, New York, NY, USA. ISBN: 0521540518 (cit. on pp. 9, 10).

[10] C. He, L. Wang, Y. Zhang and C. Wang. 2020. Dominant Symmetry Plane Detection for Point-Based 3D Models. *Advances in Multimedia*, 2020, 1, 8861367 (cit. on p. 8).

[11] Y. He, H. Huang, H. Fan, Q. Chen and J. Sun. 2021. FFB6D: A Full Flow Bidirectional Fusion Network for 6D Pose Estimation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3002–3012. DOI: 10.1109/CVPR46437.2021.00302 (cit. on pp. 5, 7, 8, 17, 26, 33, 40, 59).

[12] Y. He, W. Sun, H. Huang, J. Liu, H. Fan and J. Sun. 2020. PVN3D: A Deep Point-Wise 3D Keypoints Voting Network for 6DoF Pose Estimation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11629–11638. DOI: 10.1109/CVPR42600.2020.01165 (cit. on p. 8).

[13] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige and N. Navab. 2013. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In *Computer Vision – ACCV 2012*. K. M. Lee, Y. Matsushita, J. M. Rehg and Z. Hu, editors. Springer Berlin Heidelberg, Berlin, Heidelberg, 548–562. ISBN: 978-3-642-37331-2 (cit. on pp. 3, 7, 12, 13, 31).

[14] T. Hodaň, J. Matas and Š. Obdržálek. 2016. On evaluation of 6D object pose estimation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*. Springer, 606–619 (cit. on pp. 14–19, 21, 35, 36).

[15] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek and J. Matas. 2015. Detection and fine 3D pose estimation of texture-less objects in RGB-D images. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4421–4428 (cit. on p. 33).

[16] S. Hoque, M. Y. Arafat, S. Xu, A. Maiti and Y. Wei. 2021. A Comprehensive Review on 3D Object Detection and 6D Pose Estimation With Deep Learning. *IEEE Access*, 9, 143746–143770. DOI: 10.1109/ACCESS.2021.3114399 (cit. on p. 21).

[17] D. Q. Huynh. 2009. Metrics for 3D Rotations: Comparison and Analysis. *Journal of Mathematical Imaging and Vision*, 35, 2, (Oct. 2009), 155–164. DOI: 10.1007/s10851-009-0161-2 (cit. on p. 14).

[18] A. Krull, E. Brachmann, F. Michel, M. Y. Yang, S. Gumhold and C. Rother. 2015. Learning analysis-by-synthesis for 6D pose estimation in RGB-D images. In *Proceedings of the IEEE international conference on computer vision*, 954–962 (cit. on p. 33).

[19] T. Kurita. 2019. Principal component analysis (PCA). *Computer vision: a reference guide*, 1–4 (cit. on p. 8).

[20] A. Morawiec. 2013. *Orientations and Rotations: Computations in Crystallographic Textures*. Springer Berlin Heidelberg. ISBN: 9783662091562 (cit. on p. 18).

[21] A. Palazzolo. 1976. Formalism for the rotation matrix of rotations about an arbitrary axis. *American Journal of Physics*, 44, 1, 63–67 (cit. on p. 18).

[22] K. Pearson. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2, 11, 559–572 (cit. on p. 8).

[23] N. Pereira and L. A. Alexandre. 2020. MaskedFusion: Mask-based 6D object pose estimation. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 71–78 (cit. on pp. 17, 26, 33, 40).

[24] J. Redmon, S. Divvala, R. Girshick and A. Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. (2016). arXiv: 1506.02640 [cs.CV] (cit. on p. 3).

[25] R. Rios-Cabrera and T. Tuytelaars. 2013. Discriminatively trained templates for 3d object detection: A real time scalable approach. In *Proceedings of the IEEE international conference on computer vision*, 2048–2055 (cit. on p. 33).

[26] C. Rocchini, P. Cignoni, C. Montani, P. Pingi and R. Scopigno. 2001. A low cost 3D scanner based on structured light. In *computer graphics forum* number 3. Vol. 20. Wiley Online Library, 299–308 (cit. on p. 5).

[27] S. Rusinkiewicz and M. Levoy. 2001. Efficient variants of the ICP algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 145–152. DOI: 10.1109/IM.2001.924423 (cit. on p. 5).

[28] P. H. Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31, 1, (Mar. 1966), 1–10. DOI: 10.1007/BF02289451 (cit. on p. 7).

[29] M. Stoiber, M. Elsayed, A. E. Reichert, F. Steidle, D. Lee and R. Triebel. 2023. Fusing Visual Appearance and Geometry for Multi-modality 6DoF Object Tracking. (2023). arXiv: 2302.11458 (cit. on pp. 17, 26, 33, 40).

[30] M. Stoiber, M. Sundermeyer and R. Triebel. 2022. Iterative corresponding geometry: Fusing region and depth for highly efficient 3d tracking of textureless objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6855–6865 (cit. on pp. 17, 26, 33, 40).

[31] A. Tejani, D. Tang, R. Kouskouridas and T.-K. Kim. 2014. Latent-class hough forests for 3d object detection and pose estimation. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*. Springer, 462–477 (cit. on p. 33).

[32] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei and S. Savarese. 2019. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3343–3352 (cit. on pp. 17, 26, 33, 40).

[33] Z. Wang, X. Sun, H. Wei, Q. Ma and Q. Zhang. 2023. Enhancing 6-DoF object pose estimation through multiple modality fusion: a hybrid CNN architecture with cross-layer and cross-modal integration. *Machines*, 11, 9, 891 (cit. on pp. 17, 26, 33, 40).

[34] B. Wen, C. Mitash, B. Ren and K. E. Bekris. 2020. se (3)-tracknet: Data-driven 6d pose tracking by calibrating image residuals in synthetic domains. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 10367–10373 (cit. on pp. 17, 26, 33, 40).

[35] P. Wohlhart and V. Lepetit. 2015. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3109–3118 (cit. on p. 33).

[36] Y. Xiang, T. Schmidt, V. Narayanan and D. Fox. 2017. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *CoRR*, abs/1711.00199. arXiv: 1711.00199 (cit. on pp. 3, 4, 11, 17, 20, 22, 26, 33, 40).

[37] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai and R. Yang. 2019. IoU Loss for 2D/3D Object Detection. In *2019 International Conference on 3D Vision (3DV)*, 85–94. DOI: 10.1109/3DV.2019.00019 (cit. on p. 15).