humantech

D5.4 – Scientific report on unseen object class and shape estimation for robotic grasping

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement n° 101058236. This document reflects only the author's view, and the EU Commission is not responsible for any use that may be made of the information it contains.

Project Title	Human-Centred Technologies for a Safer and Greener European Construction Industry.
Project Acronym	HumanTech
Grant Agreement No	101058236
Instrument	Research & Innovation Action
Topic	HORIZON-CL4-2021-TWIN-TRANSITION-01-12
Start Date of Project	June 1, 2022
Duration of Project	36 months

Name of the Deliverable	Scientific report on unseen object class and shape estimation for robotic grasping
Number of the Deliverable	D5.4 (D21)
Related WP Number and Name	WP5 – Construction Robotics and Human-Robot Collaboration
Related Task Number and Name	T5.4 – Advanced Perception for Robots
Deliverable Dissemination Level	PU-Public
Deliverable Due Date	30.November 2023
Deliverable Submission Date	30.November 2023
Task Leader/Main Author	Sai Srinivas Jeevanandam (DFKI)
Contributing Partners	DFKI, SINTEF, RPTU, ACCIONA, BAUBOT
Reviewer(s)	Dr. Gabor Sziebig (SINTEF), Dr. Bruno Mirbach (DFKI), Dr. Jason Rambach (DFKI)

Keywords
Robotic perception, object pose estimation, robotic grasping, computer vision, machine learning, 6DoF pose.

Revisions

Version	Submission date	Comments	Author
v1.0	30-11-2023	First version	Sai Srinivas Jeevanandam (DFKI)

Disclaimer

Acronyms and definitions

Acronym	Meaning
DoF	Degrees of Freedom
WP	Work Package
3D, 2D	3 Dimensional, 2 Dimensional
GT	Ground truth
PnP	Perspective-n-Point
RANSAC	Random sample consensus
CNN	Convolutional Neural Networks
RoI	Region of Interest
YOLO	You Only Look Once
BOP	Benchmark on 6D Object Pose Estimation
PBR	Physically Based Rendering
IMU	Inertial Measurement Unit
AP	Average Precision
AR	Average Recall
mAP	Mean Average Precision
IoU	Intersection over Union
VSD	Visible Surface Discrepancy
MSSD	Maximum Symmetry-Aware Surface Distance
MSPD	Maximum Symmetry-Aware Projection Distance
ADD	Average Distance of Model points for Directly estimated pose
ADD-S	Average Distance of Model points for Directly estimated pose for symmetric objects

Abstract

This deliverable describes the development of robotic perception algorithms for object pose estimation in HumanTech. Object Pose estimation based on camera images as input is a key-task for localizing objects for robotic grasping. We first provide an overview of the object pose estimation problem and the overall context of the task in the HumanTech project with respect to construction material robotic grasping. Subsequently, we describe the selected object pose estimation framework for the task, the state-of-the-art algorithm ZebraPose developed at DFKI. Finally, we describe the object pose estimation task on the HumanTech object of interest category, construction bricks. We detail the approach for generating and training our machine learning models exclusively on synthetic data and conclude with an evaluation of the brick grasping pose accuracy and the next steps for integration of the method on the human-tech robotic platform for real-time functionality.

The HumanTech project

The European construction industry faces three major challenges: increase the safety and wellbeing of its workforce, improve its productivity, and become greener, making efficient use of resources.

To address these challenges, HumanTech proposes to develop **human-centred cutting-edge technologies** such as wearables for workers' safety and support and robots that can harmoniously coexist with human workers while contributing to the ecological transition of the sector.

HumanTech aims to achieve major advances in cutting-edge technologies that will enable a safe, rewarding, and digital work environment for a new generation of highly skilled construction workers and engineers.

These advances will include:

- **Robotic devices equipped with vision and intelligence** that allow them to navigate autonomously and safely in highly unstructured environments, collaborate with humans and dynamically update a semantic digital twin of the construction site in which they are.
- **Smart, unobtrusive workers protection and support equipment**. From exoskeletons activated by body sensors for posture and strain to wearable cameras and XR glasses that provide real-time workers' location and guidance for them to perform their tasks efficiently and accurately.
- An entirely new breed of **Dynamic Semantic Digital Twins (DSDTs) of construction sites** that simulate in detail the current state of a construction site at the geometric and semantic level, based on an extended Building Information Modelling (BIM) formulation that contains all relevant structural and semantic dimensions (BIMxD). BIMxDs will act as a common reference for all human workers, engineers, and autonomous machines.

The **HumanTech consortium** is formed by 22 organisations — leading research institutes and universities, innovative hi-tech SMEs, and large enterprises, construction groups and a construction SME representative — from 10 countries, bringing expertise in 11 different disciplines. The consortium is led by the German Research Centre for Artificial Intelligence’s Augmented Vision department.

Table of Contents

List of Figures

1. Introduction

Human robot Interaction is pivotal in achieving the vision of the HumanTech project. The prior deliverable *D1.1* went into the details of the HumanTech vision. Robotic devices equipped with vision and intelligence are fundamental to safely navigate the dynamic and unstructured nature of construction environments. Intelligent and Autonomous robots designed for construction sites will assist human workers, such as by transporting or discarding materials. These robots are intended to collaborate with human teams and autonomously handle routine tasks like material movement. Human-robot collaboration encompasses two principal elements: 1) human-robot communication and 2) environment perception. The human-robot communication facet guarantees that the robot comprehends instructions from the human while simultaneously offering insights to the human about its forthcoming activities. Environment perception involves using sophisticated Computer Vision algorithms to understand and infer surroundings of the robot.

Because of its dynamic nature, construction environments pose some unique challenges. The deliverable *D1.1* lists some of the main challenges and motivates the dire necessity of human robot interaction in the construction industry. Two of the main challenges observed are: 1) The physically demanding nature of the construction workplaces i.e., manual material handling, bad work postures etc., lead to multiple ailments causing long term effects and 2) increasing labour shortage in the European construction industry.

In line with this, the *Work Package 5 (WP5)* was created which is intended to deal with all tasks related to construction robots and Human robot interaction. Specifically, the *task 5.4* deals with advance perception with robots. One of the subtasks in this relates to a perception and a localization module that estimates high quality, real-time locations of the brick in the 3-dimensional world. This information allows a robot to automatically grab the brick and hand it over to the human worker.

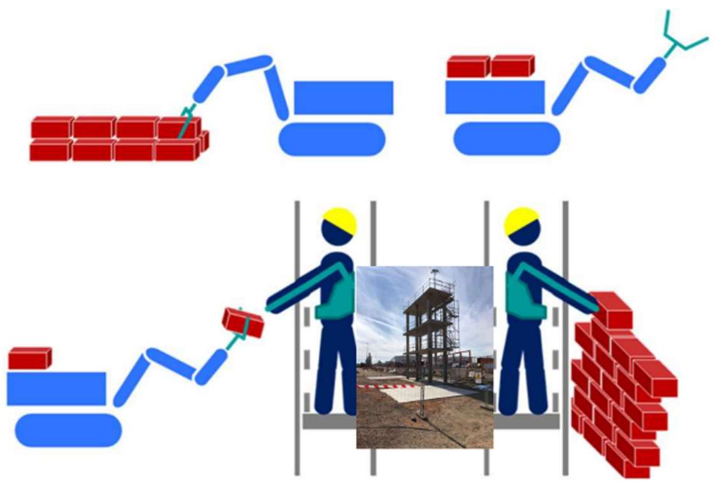


Figure 1 – Automatic brick grasping and handover Pilot.

This perception module was developed using a state-of-the-art deep learning algorithm which was developed inhouse at the *DFKI*. This module will be optimized and implemented in an embedded processing platform. The processing unit will be integrated on a mobile robotic system developed for the HumanTech project, provided by partner *Baubot*. Along with the processing platform an RGB-D sensor (Intel® RealSense™ D435i camera) and a gripper will be installed on a robotic arm of the mobile robotic system. The automatic grasping and hand-over of bricks is being developed in collaboration with our partner SINTEF. This brick grasping and transportation task will be demonstrated as part of the pilot in WP7.

1.1. Baubot robotic system

The Baubot mobile robotic system developed for the HumanTech project is a mobile robotic platform equipped with an industrial collaborative robot manipulator. The robotic system is equipped with various sensor systems that help it in its decision-making process. The robotic system is built in a modular way where new application use cases can be developed independently and integrated into it. This aids in rapid research and development along with parallel development cycle for various challenging use cases.

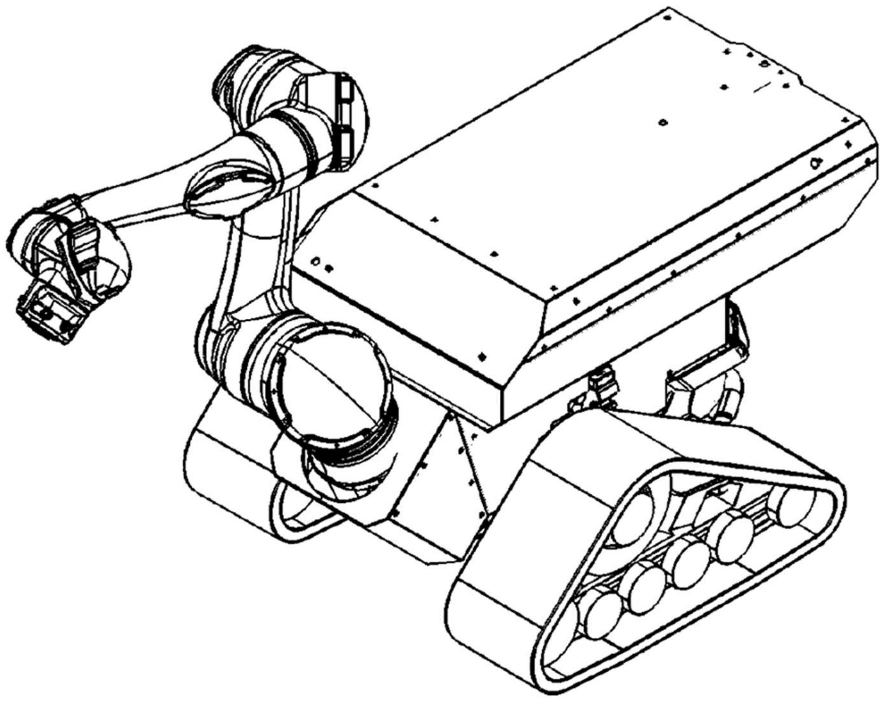


Figure 2 – Baubot Mobile robotic platform.

Figure 2 shows the Baubot mobile robotic platform with its basic elements. Applications involving human robot collaboration can be additionally developed and integrated into the platform. The robotic system can take inputs from various external applications and perform actions based on the commands that are being passed to the robot.

1.2. Gripper and camera on the human-robot arm from SINTEF

Figure 3 shows a 3D-model of the brick gripper. The gripper is meant to grasp a brick from a pallet, transfer it safely to a position that is ergonomically favourable for the human worker, and finally hand it over.

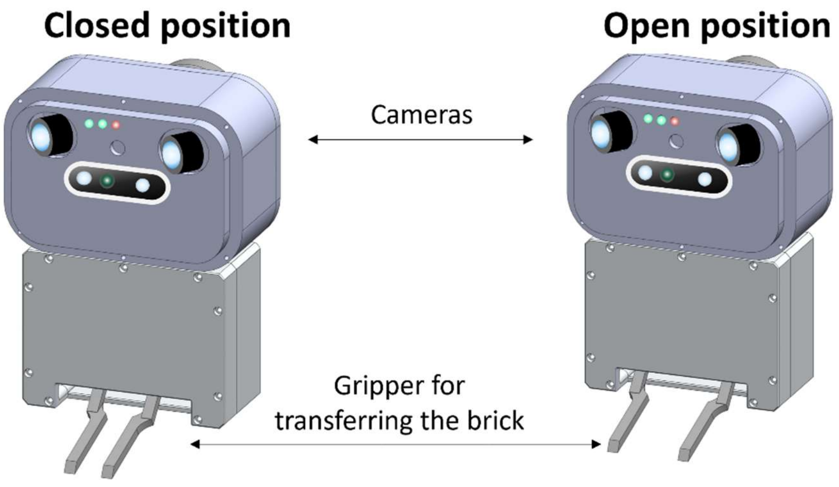


Figure 3 – Gripper with cameras for handing over bricks form a pallet to a worker.

The sensor unit above the gripper is attached to the robot flange. The main purpose of the unit is to house the sensors, including a camera used for brick identification. In this way, the robot, via manipulation, can bring the sensor to the image acquisition location such that the brick is in the field of vision and at the desired distance. Once the brick is identified and located, the robot can perform manipulation for gripping it.

Powered by a motor on the backside of the gripper, the two fingers move together and apart simultaneously. In the closed position (Figure 3 left) the fingers can be moved into the cavities of the brick. Afterwards, the fingers will be moved to the open position (Figure 3 right) and secure the brick by applying a constant torque and thereby apply a clamping force to the inner walls of the cavities. While handing over the brick, the fingers move toward the closed position, releasing the brick, which then hangs freely on the fingers and can be taken by a human worker.

1.3. 6 Degrees of Freedom (6DoF) Object Pose Estimation

Object Pose Estimation refers to the task of determining the spatial orientation and position of an object within an image or a video sequence. This is crucial in understanding the three-dimensional (3D) configuration of objects in two-dimensional (2D) images. Object pose estimation is a pivotal computer vision task with significance in numerous fields, such as **Robotics**, by enabling robots to perceive and manipulate objects effectively; **Augmented Reality,** for positioning virtual objects accurately in the real world; **Quality Control**, to check the orientation and alignment of parts in manufacturing; and **Autonomous Vehicles**, for identifying and navigating around objects.

1.3.1. Definition of 6DoF Object Pose

6DoF object pose refers to determining an object's position i.e., translation along the three perpendicular axes (*X*, *Y*, and *Z*) and orientation i.e., the rotation along these axes. These are often described by yaw, pitch, and roll. The 6DoF corresponds to three translational and three rotational parameters, which jointly describe an object's overall pose with respect to a camera in the 3D space. Therefore, the transformation of a 3D point on an object from the object coordinate system O to the camera coordinate system C is given by the 6DoF pose.

Formally, the conversion from a coordinate system A to a different coordinate system B is a 6DoF transform consisting of a rotation matrix $\boldsymbol{R_{ba}} \in SO(3)$ and a translation vector $a_b \in R^3$. The rotation matrix R_{ba} describes the rotation from coordinate system A to B whereas a_b is the origin of coordinate system B expressed in coordinate system A. Now, transformation of a point between the coordinate systems is given as

$$\boldsymbol{m_b} = \boldsymbol{R_{ba}}\boldsymbol{m_a} + \boldsymbol{a_b} \qquad \text{(i)}$$

Using this notation, the conversion of a 3D point $\mathrm{m_o} = [\mathrm{x_o}, \mathrm{y_o}, \mathrm{z_o}]^{\mathrm{T}}$ in the object coordinate system O to the 3D point $\mathrm{m_c}$ in the camera coordinate system C can be written as shown in the Equation (ii). Figure 4 also gives a graphical illustration of this transformation.

$$\boldsymbol{m_c} = \boldsymbol{R_{co}}\,\boldsymbol{m_o} + \boldsymbol{o_c} \qquad \text{(ii)}$$

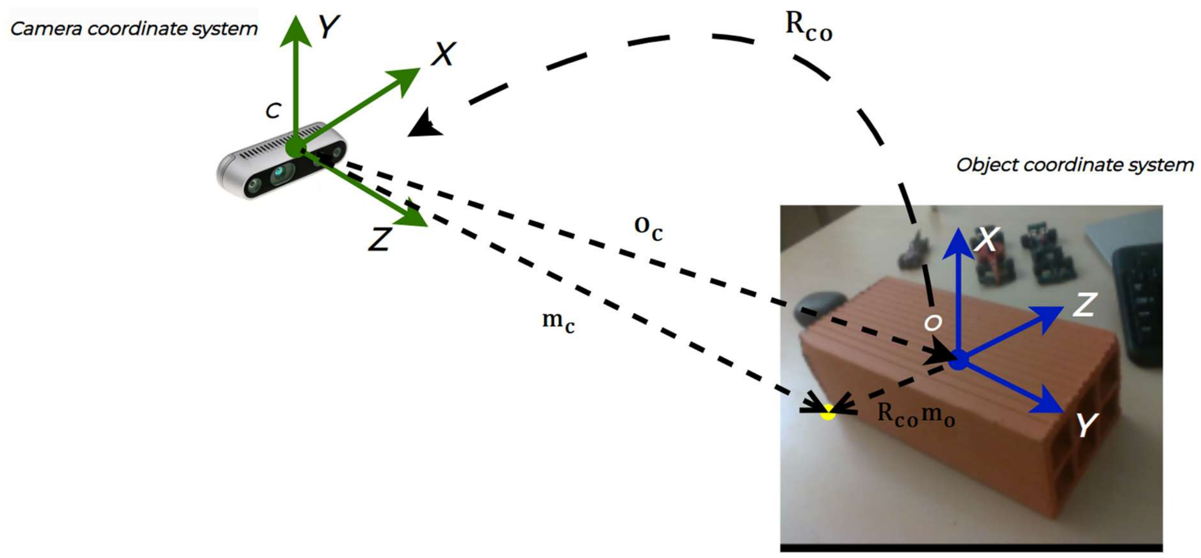


Figure 4 – Illustration of 6DoF pose estimation.

Determining the poses of objects enhances both the semantic and geometric comprehension of a scene. This enables a computer vision system to recognize not only what is in its view but also the precise 3D location of objects in relative to the camera. Such information is vital for numerous computer vision applications. Moreover, retrieving an object's 3D placement inherently provides insights into its 2D location and its 2D segmentation mask.

6DoF pose estimation is a fundamental problem, bridging computer vision and robotics, enabling machines to perceive and interact with their environment in a 3D space. It combines spatial geometry and deep learning insights, aiming to deal with the complexities and variabilities in real-world scenarios. Ongoing research focuses on enhancing accuracy, robustness, and generalization across different objects and environments, driving advancements in robotics, Augmented Reality (AR)/Virtual Reality (VR), and more.

1.3.1.1 Applications

The importance of 6DoF pose estimation can be seen in many applications. Three main examples of this are:

- **Manipulation**: In robotics, accurate 6DoF pose estimation is crucial for manipulation tasks like picking and placing objects.
- **Navigation**: For drones and autonomous vehicles, understanding the pose of obstacles and entities is critical for navigation.
- **AR/VR**: Precise 6DoF tracking of devices and entities ensures immersive and realistic AR/VR experiences.

1.3.2. Challenges

6 DoF pose estimation, especially. Pose estimation form a single RGB image is an inherently challenging task. Some of the challenges in 6DoF pose estimation are:

- **Lack of quality, real training data.** A primary challenge for learning-driven pose estimation techniques is the complexity of labelling real training datasets with pose information. Labelling images with ground truth (GT) object poses is an extremely tedious and intricate task. The quality of these manual annotations is far from perfect. This is bad for learning based methods. Therefore, lot of **deep learning** based methods use *synthetically generated data* for training and use the real data to test and perform inference.
- **Projection ambiguity.** Pose estimation, when derived directly from single monocular RGB images, is notably a critical issue due to its potential for widespread applications. Observing the approaches employed in the BOP challenge [1]–[3], most contemporary methods grounded in deep learning particularly concentrate on this situation for preliminary pose estimation. Techniques utilizing depth or multiple views are typically applied merely for subsequent refinement. Nonetheless, the inherent challenge of monocular pose estimation lies in its ambiguity, as the projective transformation and the discrete structure of images complicate the precise determination of an object's distance on the camera's z-axis.

- **Real time performance.** For pose estimation to have practical value, it needs to operate in real-time and maintain high precision. Balancing these two requirements is notably difficult. While direct pose regression techniques can be quick, they often compromise on accuracy [4]–[6]. In contrast, hybrid methods based on correspondence generally provide superior pose accuracy but are slower [7]–[9]. The complexity increases when multiple objects are present in the scene, as they are usually processed individually.
- **Occlusion**: Partial visibility or complete invisibility of objects makes accurate estimation difficult.
- **Symmetry**: Objects with rotational symmetries present challenges in determining the correct orientation. Symmetries create a problem due to the ambiguities in training caused by different ground truth for same visual pose.
- **Scale**: Variations in object size and distance from the camera affect the model's perception and prediction.
- **Lighting and Shadows**: Variability in illumination and shadowing can mislead estimations.
- **Texture**: Texture-less or highly textured objects can skew predictions due to the absence or overabundance of features.

1.4. Related work

6DoF pose estimation has become a fundamental computer vision problem [10], [11]. Until the advent of deep learning, traditional methods which involved hand crafted features were the norm. These have been then replaced by learning-based methods which involved Convolutional Neural Networks (CNNs). Research on 6DoF pose estimation can be broadly categorised as follows:

1.4.1. Traditional Methods

Traditional methods mainly involve using geometric and feature based methods. Pre-deep-learning, researchers proposed a variety of hand-crafted algorithms to detect key points and generate features. Some examples of this are SIFT [[12], SURF [13], BRIEF

[14] etc. Using the key points and features, the object pose problem can be estimated by forming 2D-3D correspondences between the input image and the 3D object model. The 2D-2D correspondences are then used by some geometric algorithms to detect pose. These are as follows:

PnP (Perspective-n-Point) [15] **with Random sample consensus (RANSAC)** [16]: Given a set of 2D image points and their corresponding 3D world points, the pose of the camera is determined using the PnP algorithm. RANSAC will then estimate the poses more robustly by filtering out outliers. PnP can be used as a standalone module, but usually PnP is always used with in a RANSAC framework for outlier rejection.

Stereo Vision: Triangulates 3D point positions using 2D points from two calibrated cameras to aid in pose estimation.

1.4.2. Template and Retrieval-Based Methods

The fundamental concept behind template-based techniques involves using a collection of images, termed as templates, that depict the object from multiple angles. This collection is designed to encompass nearly all conceivable views of the object. During evaluation, the input image's representation is assessed against this entire collection based on a specific similarity metric. The top matches are then refined to produce the final pose. One of the leading pre-deep learning template-based methods for 6 DoF pose estimation is LineMOD [17]. SSD6D detector [18] is a popular end-to end deep learning solution based on the template matching idea.

1.4.3. Direct Pose Regression

Neural networks can also be used directly to predict a mapping from an input image to the object pose. The work by Rambach, Jason, et al., *"Learning 6dof object poses from synthetic single channel images."* [19], was one of the first methods where a CNN is utilized to regress the position and orientation of a camera with respect to an object given an RGB image. Subsequently, other methods like PoseCNN [4] and DenseFusion [5] also proposed RGB based pose estimation. More recently, CosyPose [6] revisited the task and proposed a 3-stage deep learning based approach and utilises synthetic data for better understanding. Another recent work is GDR-Net [20] which uses geometry-guided direct regression network for monocular 6D object pose estimation.

Even though direct pose regression methods appear straightforward to implement; they often yield lower accuracy compared to template- and correspondence-based solutions.

1.4.4. Hybrid approaches

Recently, combining learning-based approaches with geometric methods are showing great results when compared to the weak accuracies of pure end-to-end learning methods. AAE [21] learns to generate a latent vector based on the visual information of the object in discrete viewpoints. At inference stage, the rotation is obtained by comparing the latent code with the pre-generated rotation-latent code lookup table. Other indirect/hybrid methods usually estimate the 2D-3D correspondence, and solve the object pose using RANSAC/PnP. BB8 [7] firstly defines the 3D object bounding box corners as the keypoints and PVNet [22] reaches high recall rate by predicting the keypoints with a dense pixel-wise voting for sampled keypoints on the object. The main drawback of such sparse 2D-3D correspondence methods is that the prediction of keypoints in the occluded area lacks in accuracy. Another alternative approach involves leveraging dense correspondences, wherein every visible object pixel in the image is mapped to its 3D counterpart. The belief is that by increasing the count of these correspondences, potential inaccuracies can be offset, yielding more accurate poses. Whereas having a limited number of correspondences might produce less accurate poses, a comprehensive set of correspondences has the potential to deliver better results. Additionally, this approach offers an improved way to handle occlusions. The downside is that, because of the dense correspondences, these methods are slow. iPose [8] which operates in 3 stages: segmentation, 3D coordinate regression and pose estimation, Pix2Pose [9] which unifies the first two stages also into an end-to-end network, are two examples of this.

2. Object Pose Estimation algorithm – ZebraPose [23]

ZebraPose [23], is a single RGB-based 6DoF pose estimation algorithm. ZebraPose proposes a dense correspondence pipeline that combines the concepts of handcrafted features and image segmentation in a hierarchical fashion for RGB based 6DoF pose estimation. To design a descriptor that encodes surfaces efficiently, ZebraPose uses the binary numeral system. Binary-based descriptors are applied other computer vision applications like SLAM [24]. In ZebraPose the object surface is split into halves or groups over multiple iterations and an encoding of each vertex is done by stacking the assigned group labels. By leveraging a hierarchical discrete representation, ZebraPose guarantees a compact mapping and simple learning objective as a multi-label classification problem. Moreover, learning how to encode a full sequence at once might be challenging for neural networks. Therefore, in this algorithm a coarse to fine learning scheme is utilized. By design, the encodings on the coarse levels are continuously shared in wider object regions. As the network learns to differentiate coarse splits, it focuses on finer encoding positions. With a coarse to fine loss and training strategy, ZebraPose then manages to predict fine-grained surface correspondences. In contrast to previous works, ZebraPose encoding mechanism promotes direct pixel-to-surface matching just by means of a look-up table. With a simple matching and PnP-RANSAC, it outperforms the state of the art in 6DoF pose on the most commonly used benchmarks.

In summary, ZebraPose algorithm is a two-stage RGB-based approach that defines the matching of dense 2D-3D correspondence as a hierarchical classification task. Further, the two-stage algorithm has three components:

1. Assigning a unique descriptor to the 3D vertex called *3D surface code generation*.
2. Predicting a dense correspondence between the 2D pixels and 3D vertices.
3. Solving the object pose using the predicted correspondences.

2.1. 3D Surface Code generation

Each vertex on the 3D surface of the object is encoded with a fixed length vertex code. This encoding is defined based on the vertices position relative to the 3D surface to enable coarse to fine learning. To enable this, the encoding is constructed with a non-decimal numerical system with lower radix. This ensures that the representation is very efficient. For a code of length *d*, *d* iterations of grouping of the vertices are performed. Finally, each vertex is assigned to a vertex code with d digits by stacking the class id of each grouping operation. This representation is stored and fixed for every 3D object. The vertices in each group share the same code. A lookup table is built to map a code to the centroid of each group in G_d, which is further used to build 2D-3D correspondence and solve the pose. By means of experimenting and ablation studies, a binary system is selected as the base for the vertex code. This makes the grouping easier to split i.e., black (1) and white (0). The binary descriptor allows one-to-one correspondence for the problem of 6DoF object pose efficiently and leads to a big advantage in reducing the GPU memory requirement, thus making the whole process faster. This process happens offline and the generated mapping between vertex code and the corresponding 3D vertex is stored in a look-up table. The illustration for this surface code generation is illustrated in the below Figure 5.

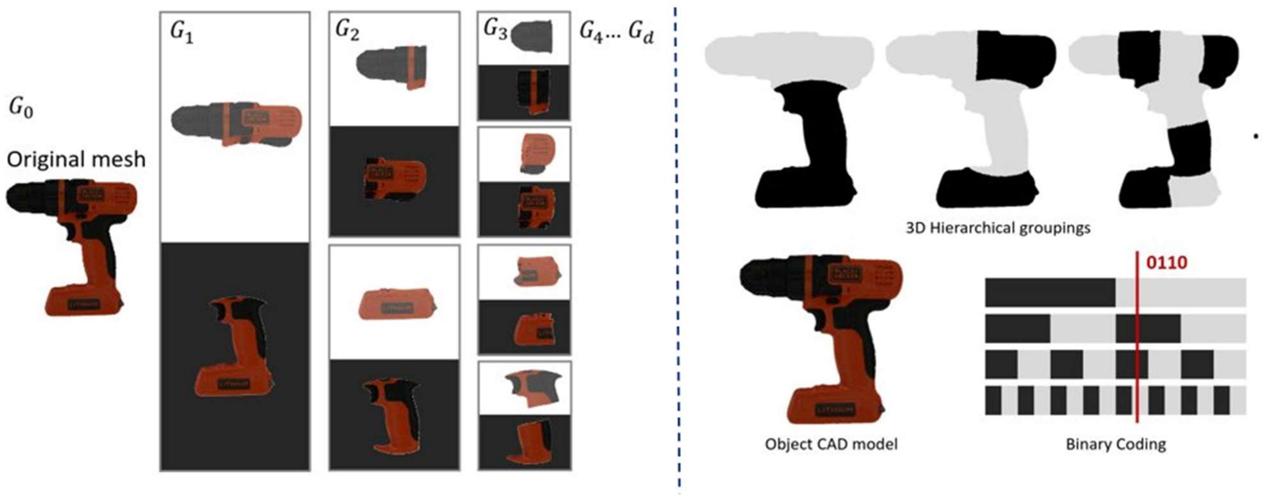


Figure 5 – ZebraPose Surface code generation [23]

2.2. Pose Estimation Network

To predict the code per pixel in the frame with fine granularity, only the Region of Interest (ROI) around object pixels is considered. This ROI is predicted by a 2D detector.

The predictions from the 2D detector are cropped and resized from the image to form the ROIs. The target vertex code maps are generated using a Convolutional Neural Network (CNN) based encoder-decoder network. Using the predicted code and the visible mask from the CNN, a correspondence matching is performed. For this matching, first the look-up table generated before is used to extract corresponding 2D and 3D points. Following that, a RANSAC/PnP solver is used to calculate the rotation *R* and translation *t* of the object relative to the camera. The entire process from input images to the predicted pose from the ZebraPose architecture is presented in the Figure 6.

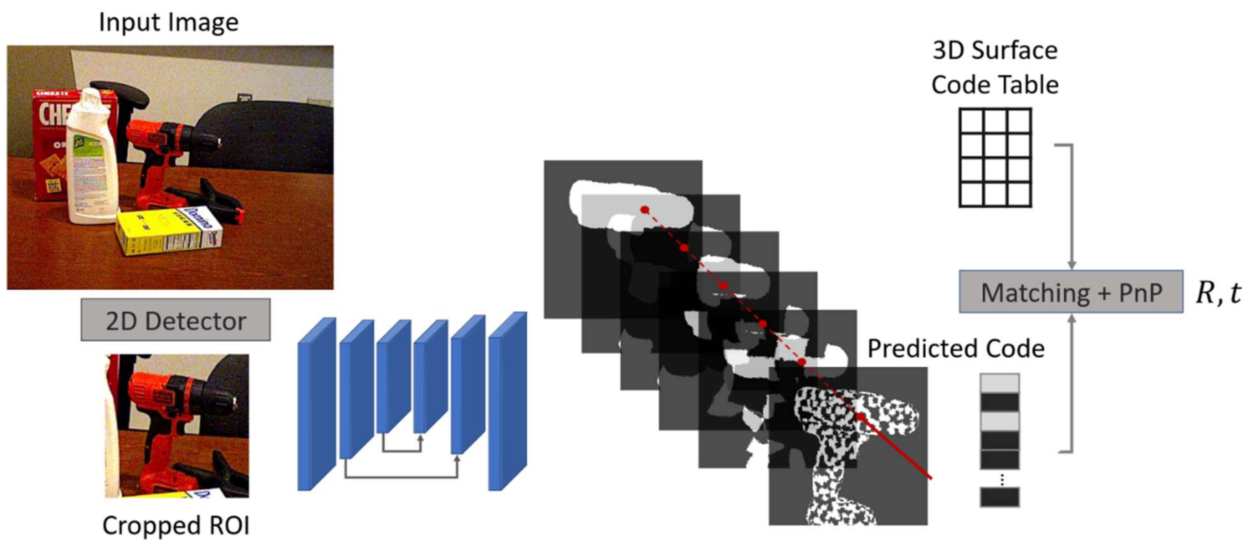


Figure 6 – ZebraPose Architecture [23]

2.3. 2D detection network – YOLOv8

As depicted in the Figure 6, the CNN estimates the pose with the cropped ROI from a detected bounding box. The object pose estimation is meaningless with a false-positive detection, also the pose is not even estimated in the case of false-negative detection. Therefore, having a 2D detector with high recall and precision is important to leverage the detected bounding boxes.

As a 2D detector we modified the state-of-the-art real-time 2D detector called YOLOv8 for our purposes. YOLOv8 [25] is a cutting-edge, state-of-the-art model that builds upon the success of previous You Only Look Once (YOLO) versions and introduces new features and improvements to further boost performance and flexibility. YOLOv8 is designed to be fast, accurate, and easy to use, making it an excellent choice for a wide range of object detection and tracking, instance segmentation tasks. YOLOv8 is built on cutting-edge advancements in deep learning and computer vision, offering unparalleled performance in terms of speed and accuracy. Its streamlined design makes it suitable for various applications and easily adaptable. The architecture of YOLOv8 is show in the below.

.

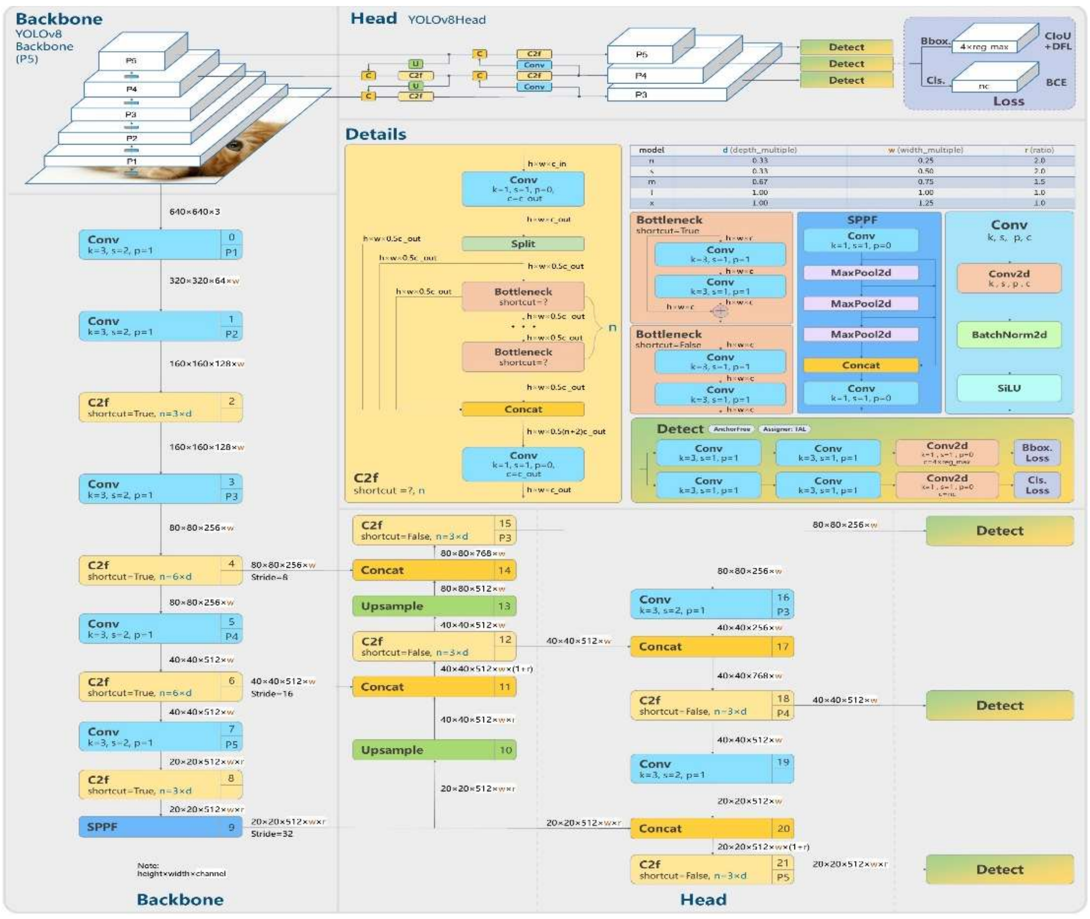
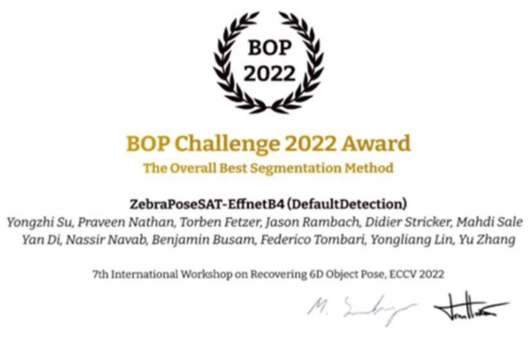


Figure 7 – YOLOv8 architecture [25]

2.3. Benchmark on 6D Object Pose Estimation Challenge and Awards

The BOP (Benchmark on 6D Object Pose Estimation) Challenge [1]–[3] is a significant benchmark in the field of computer vision, specifically for object pose estimation. The BOP Challenge is an annual competition that seeks to evaluate and compare state-of-the-art methods for 6DoF object pose estimation. It provides standardized datasets, evaluation metrics, and tools to ensure consistent comparison of different algorithms. The BOP Challenge employs multiple evaluation metrics, to evaluate the performance of the algorithms (Discussed more in Section 3.4). These metrics give a detailed insight into how well algorithms perform in terms of both rotation and translation estimations. The BOP challenge is significant in **Bridging Research and Application, Understanding Occlusions and Multiple Objects and Stimulating Progress.** The BOP Challenge plays an integral role in driving progress in 6D object pose estimation. As techniques continue to evolve and become more sophisticated, benchmarks like BOP provide a pathway, guiding researchers and practitioners towards the perfect pose estimation in any given scenario.


BOP 2022

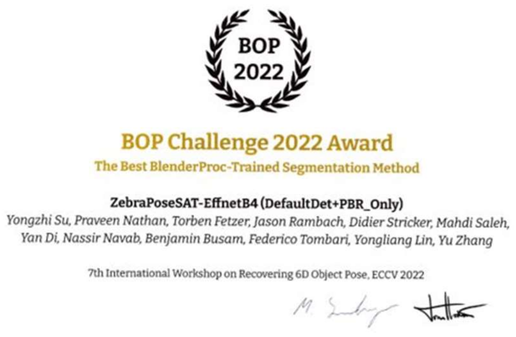
BOP Challenge 2022 Award

The Overall Best Segmentation Method

ZebraPoseSAT-EffnetB4 (DefaultDetection)

Yongzhi Su, Praveen Nathan, Torben Fetzer, Jason Rambach, Didier Stricker, Mahdi Sale Yan Di, Nassir Navab, Benjamin Busam, Federico Tombari, Yongliang Lin, Yu Zhang

7th International Workshop on Recovering 6D Object Pose, ECCV 2022


BOP 2022

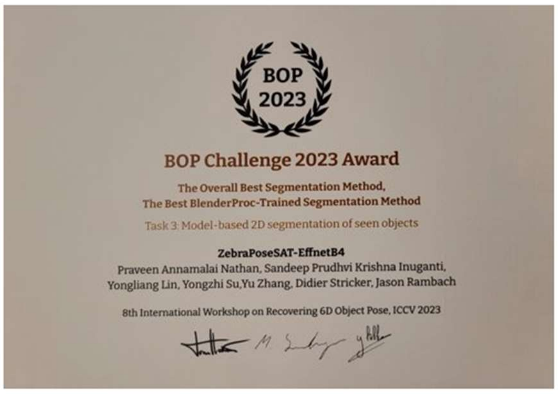
BOP Challenge 2022 Award

The Best BlenderProc-Trained Segmentation Method

ZebraPoseSAT-EffnetB4 (DefaultDet+PBR_Only)

Yongzhi Su, Praveen Nathan, Torben Fetzer, Jason Rambach, Didier Stricker, Mahdi Saleh, Yan Di, Nassir Navab, Benjamin Busam, Federico Tombari, Yongliang Lin, Yu Zhang

7th International Workshop on Recovering 6D Object Pose, ECCV 2022


BOP 2023

BOP Challenge 2023 Award

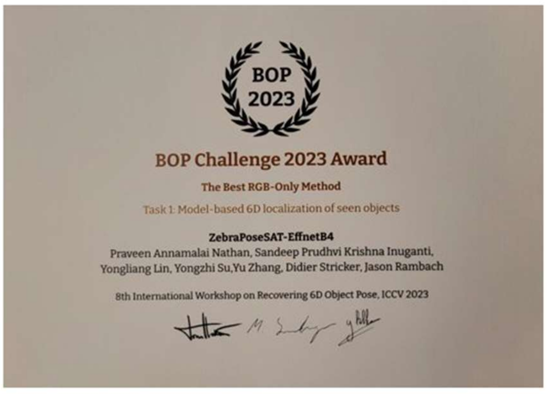
The Overall Best Segmentation Method,
The Best BlenderProc-Trained Segmentation Method

Task 3: Model-based 2D segmentation of seen objects

ZebraPoseSAT-EffnetB4

Praveen Annamalai Nathan, Sandeep Prudhvi Krishna Inuganti, Yongliang Lin, Yongzhi Su,Yu Zhang, Didier Stricker, Jason Rambach

8th International Workshop on Recovering 6D Object Pose, ICCV 2023


BOP 2023

BOP Challenge 2023 Award

The Best RGB-Only Method

Task 1: Model-based 6D localization of seen objects

ZebraPoseSAT-EffnetB4

Praveen Annamalai Nathan, Sandeep Prudhvi Krishna Inuganti, Yongliang Lin, Yongzhi Su,Yu Zhang, Didier Stricker, Jason Rambach

8th International Workshop on Recovering 6D Object Pose, ICCV 2023

Figure 8 – BOP Challenge awards

The DFKI team participated in the BOP challenge where the ZebraPose [23] algorithm was also evaluated. Compared with other methods using RGB input, ZebraPose ranked first in multiple datasets even without using pose refinement. The ZebraPose was evaluated in BOP challenge 2022 [3] and BOP challenge 2023. In both the years, the algorithm won first place and multiple awards in different categories. The total list of winners and awards are available on the website https://bop.felk.cvut.cz/leaderboards/ . Some of the awards that the DFKI team has won are shown in the Figure 8.

3. Pose Estimation for Unseen Object Robotic Grasping

Coupled with the challenges mentioned in the above sections, and the presence of numerous unseen objects in a construction environment, pose estimation becomes a difficult task. In this section, we discuss our design of an end-to-end pipeline based on a single RGB image, from a sensor mounted on a robotic arm, for 6DoF pose estimation of an unseen object i.e., a brick as an example. As discussed in the introduction section, one of the main objectives of this algorithm is to determine the poses of bricks and communicate with the robotic arm for brick laying task. The construction brick as shown in the following figures is considered for this use case.

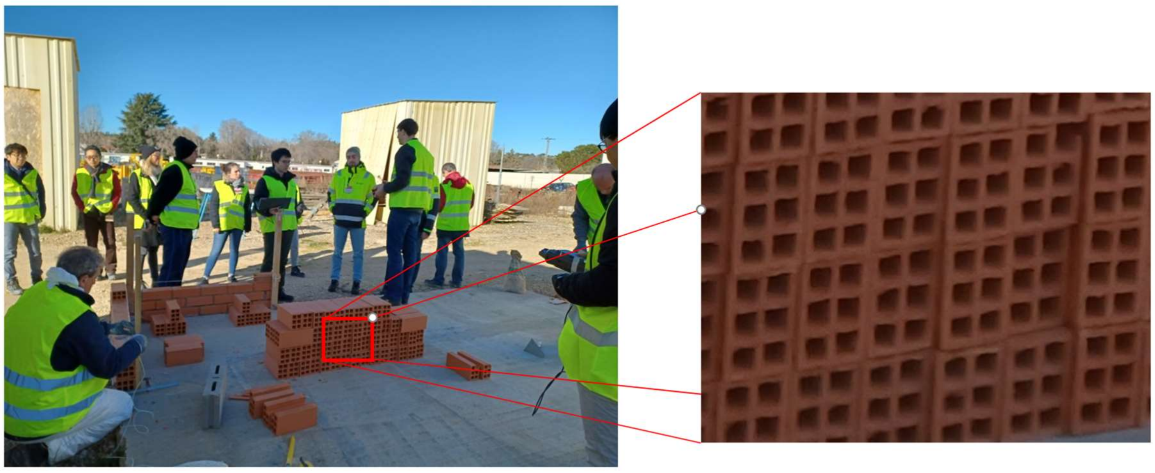


Figure 9 – Brick for pose estimation in the construction environment.

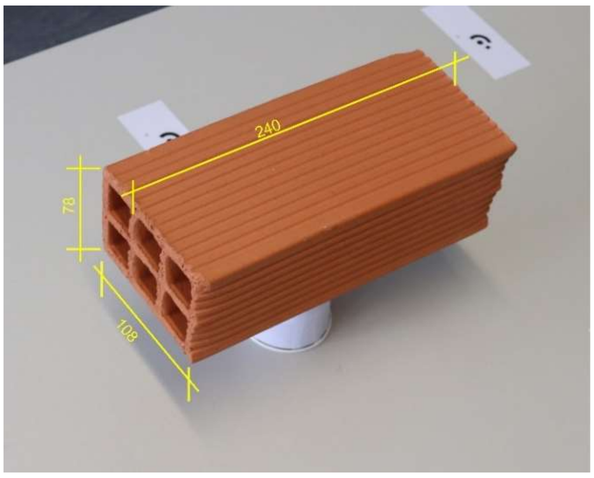


Figure 10 – Measurements of the brick.

The bricks are from the construction sites in Spain. Our partner *Acciona* facilitated the bricks to perform our experiments.

On first examination, it can seem like bricks can be generalized to a simple cuboid structure, and estimating the poses of them might be an easy task. But every brick has its own characteristics which introduce additional constraints and are needed to be taken into consideration. Also, in cluttered environments, apart from the brick, other cuboid like structures should not be detected. Some main challenges that occur in dealing with this are, **Textures,** each brick has its own texture which is very important and uniquely defines each brick. These textures allow for feature extraction from deep learning. Capturing this texture information is critical; **Symmetries,** most bricks are inherently symmetric in nature. Symmetric objects introduce ambiguity to the pose estimation task because multiple poses can appear identical in an image. These objects often necessitate extra attention during the creation of ground truth data and in defining the loss function; **Occlusions,** as shown in the Figure 9, the stack of bricks directly introduces occlusion. In many cases, we can only see one face of the brick. Detecting these highly occluded objects is an extremely difficult task. These challenges are not only for bricks but can also be generalised to any objects.

As deep learning methods learn from data, having a proper dataset with ground truth annotations is of paramount importance. However, as mentioned in the challenges section 1.3.2, creation of a real dataset with proper GT annotations is very tedious and difficult task. Having the manual GT annotations still doesn't guarantee proper quality and can result in more bad than good when training the networks. To tackle this using ***Synthetically generated*** datasets is a really good solution.

3.1. Synthetic Dataset

As an alternative to deal with the unavailability of real data, synthetically generated datasets can be used. A **synthetically generated dataset** refers to a collection of data that has been created artificially through computational and algorithmic means rather than being collected from real-world observations. In the context of deep learning and computer vision, synthetic datasets can be especially valuable due to their ability to provide a rich, annotated source of training data under various controlled conditions, which might be challenging, expensive, or ethically complicated to collect in the real world. One example of a computation tool that can be used to do this is BlenderProc [26].

BlenderProc is a popular procedural Blender pipeline, which implies it generates 3D scenes by running a sequence of modules in a predefined order. These modules can generate and manipulate 3D scenes, render images, and compute various types of data. Leveraging this tool along with 3D models can create synthetic datasets for computer vision tasks. However, it's crucial to carefully design and validate synthetic datasets to ensure they effectively aid model training and are representative of real-world scenarios. Ensuring synthetic data closely mimics real-world scenarios to allow models to generalize well is very important. Also, the neural network model should be designed such that it will not overfit easily on the data it is provided for training.

ZebraPose was designed to deal with exactly this challenge. ZebraPose generalizes well to real data even when completely trained on synthetic data.

3.1.1. Synthetic training data pipeline

To train ZebraPose model to retrieve the 6DoF pose of an unseen object, in this case our brick, we need to create a synthetic dataset.

Our partner at RPTU provided us with 3D models of the brick using very high-quality industrial level scanning. The following Figure 11 shows a visualization of 3D model of the brick.

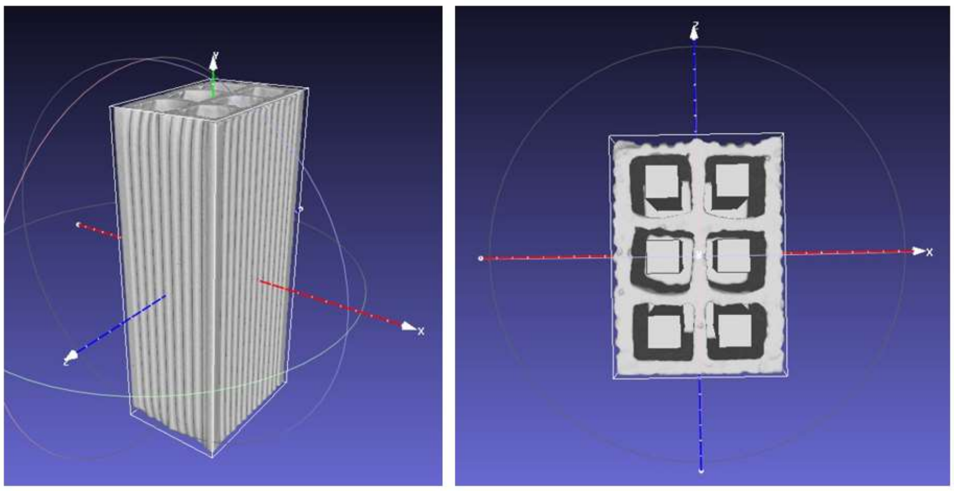


Figure 11 – 3D model of the brick

BlenderProc4BOP is a data synthesis approach, based on BlenderProc [26], implemented as part of the BOP [1]–[3] challenge, which aims at capturing the state of the art in estimating the 6D pose. Recent works [27], [28] have shown that, physically based rendering (PBR) and realistic object arrangement help to reduce the synthetic-to-real domain gap in object detection and pose estimation tasks. BlenderProc4BOP exploits these advantages to achieve a rich spectrum of generated images. In this data synthesis approach, the selected object is arranged inside an empty room, with objects from other BOP datasets used as distractors.

Keeping with the context of the brick laying task discussed before, the dataset will be created with stacks of bricks that would represent the same in a construction site. This can be seen in Figure 9. The most important point in creating a synthetic dataset is that it should capture the real-life scenarios to as close as possible. The following figures show different rendering scenarios that were considered to emulate real-life scenarios.

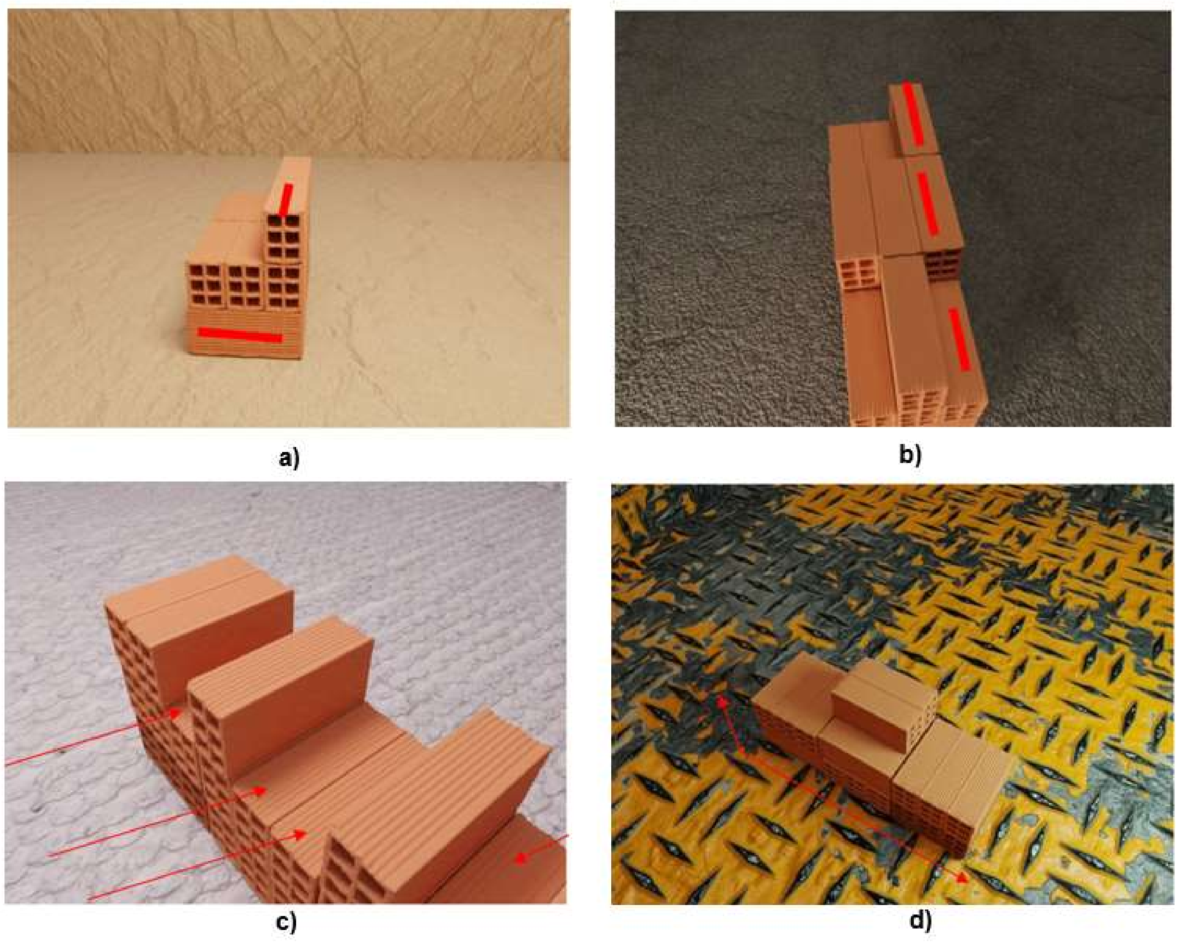


Figure 12 – a) Random orientation; b) Same orientation; c) Top layer random missing brick; d) Random length and height.

The above images already look very realistic, but they look too perfect compared to real life scenarios. To add more unevenness, we also add offsets to the bricks resulting in more realistic looking stacks. Figure 13 below depicts this.

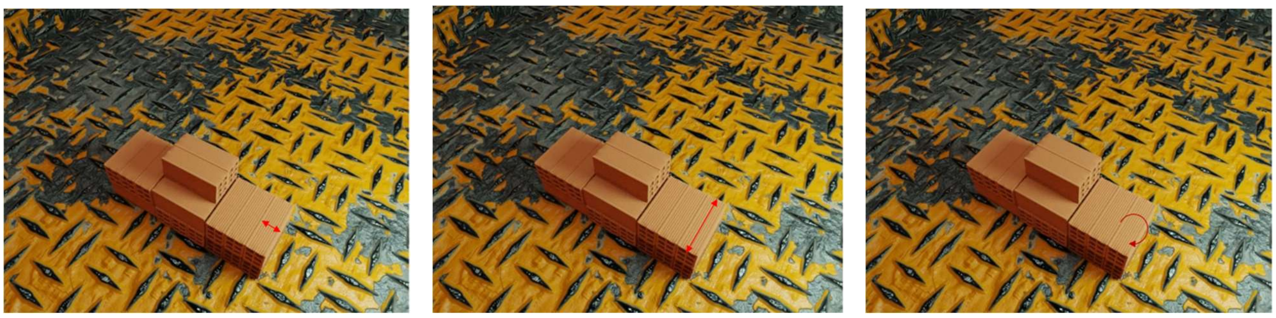


Figure 13 – Adding offsets to the bricks.

Moreover, by using varying number of backgrounds we can create a dataset which has a lot of variations to assist the networks in generalizing well. Finally, randomly placing objects from the BOP [3] datasets around the stack of bricks and randomly placing the bricks in the scene, instead of stacks of bricks, completes the dataset. The final dataset is shown in Figure 14 below. Here, the brick was taken as an example as it was also aligning with the other tasks in the HumanTech project specifically in *WP5* and *WP7*. But this can be generalized to any other unseen object and a similar photorealistic dataset can be created.

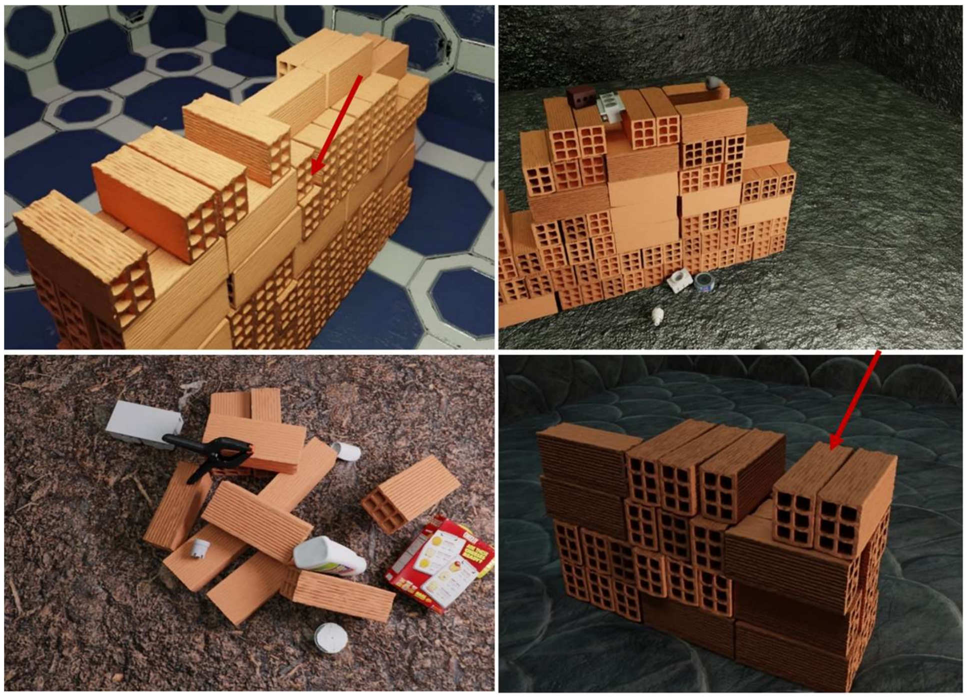


Figure 14 – Final dataset with all the scenarios.

3.1.2. Dataset properties

The final dataset consists of about 10,000 images. Although this might seem too few, our ZebraPose algorithm has shown good performance at learning correspondences and accurately estimating poses from similar amounts of images even if they are synthetically generated. The training images are divided into two types, type 1 consisting of stacks which amount to about 80 percent of the dataset and type 2 which consists of randomly placed bricks which amount to about 20 percent of the dataset. The dataset is split into three parts, i.e., *train, val* and *test.* Each split contains *9500*, *300* and *200* images respectively. The *train* and *val* splits are used during the training. The *test* split is used to evaluate the performance of the trained detector and pose estimator. The quantitative result of the evaluation is shown in Section 3.4.

3.2. Training Pipeline

Once the dataset was generated as discussed above, ZebraPose needs to be trained to retrieve the 6DoF poses. As explained before and as shown in the

Figure 6, ZebraPose also requires a region of interest from a 2D detector. The 2D detector as well as the ZebraPose algorithm was trained with the synthetically generated dataset. The trained YOLOv8 2D detector is then used to retrieve 2D detections i.e., localizing the bricks in the image. This is done in the form of detecting 2D bounding boxes. These boxes are now the region of interests which are cropped from the image and sent to the ZebraPose network. The region of interest contains the object for which the 6DoF pose needs to be retrieved.

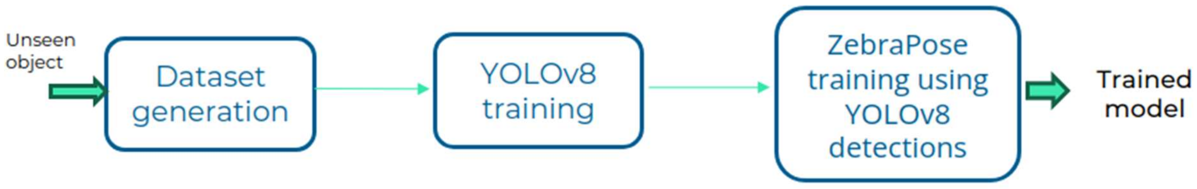


Figure 15 – Training pipeline

3.3. Realtime 6DoF pose estimation for robot grasping.

In addition to the need of more than one modality (depth, point cloud etc.), the main limitations of many 6DoF pose estimation algorithms is that they are not real time. Using the training mechanism explained in previous section, we designed an end-to-end pipeline to retrieve poses at real time. The pipeline requires just a single RGB image, and the result is the poses of the objects in this RGB image. A visualization depicting the outline of the realtime pose estimation pipeline from an RGB image in the context of brick pose is shown in the figure below.

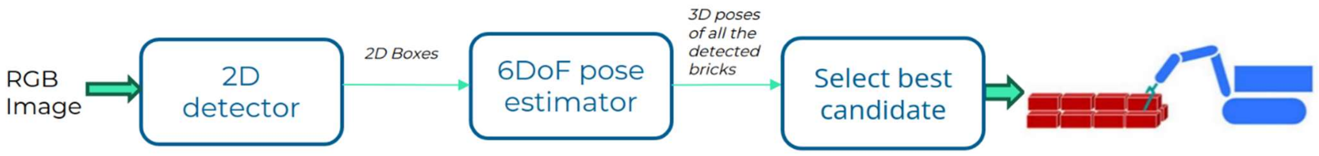


Figure 16 – Real time pose estimation pipeline for robot grasping.

In the Figure 16, the *select best candidate* module, will take the predicted 3D poses of all the detected bricks and select one or more best candidates that can be grasped or picked up by the robotic arm. In the case of brick laying task, Figure 17 shows some possible candidates that a robotic arm can grasp. The selection of the best candidate bricks will be based on three main factors 1) **Occlusion ratio,** by comparing the ratio of the Amodal mask and Modal mask. The Amodal mask refers to the mask of the object which also contains the occluded regions, and the Modal mask refers to only the visible parts of the object in the image. Therefore, by taking the ratio of these we can decide which of the bricks should be eliminated as a candidate. If this occlusion ratio is high, it means the object is occluded by other objects and surroundings in the 3D world. The objects with lowest occlusion ratios are selected as candidates. 2) **Relative height,** in the case of brick stacks, the bricks which are on the top of the stacks are an ideal option. Using the IMU sensor and the world coordinate to camera coordinate system transformations, the top bricks are identified. 3) **Network confidence,** the ZebraPose algorithm also predicts a confidence value for each pose estimation. This confidence value is also considered as a filter to select the best candidate.

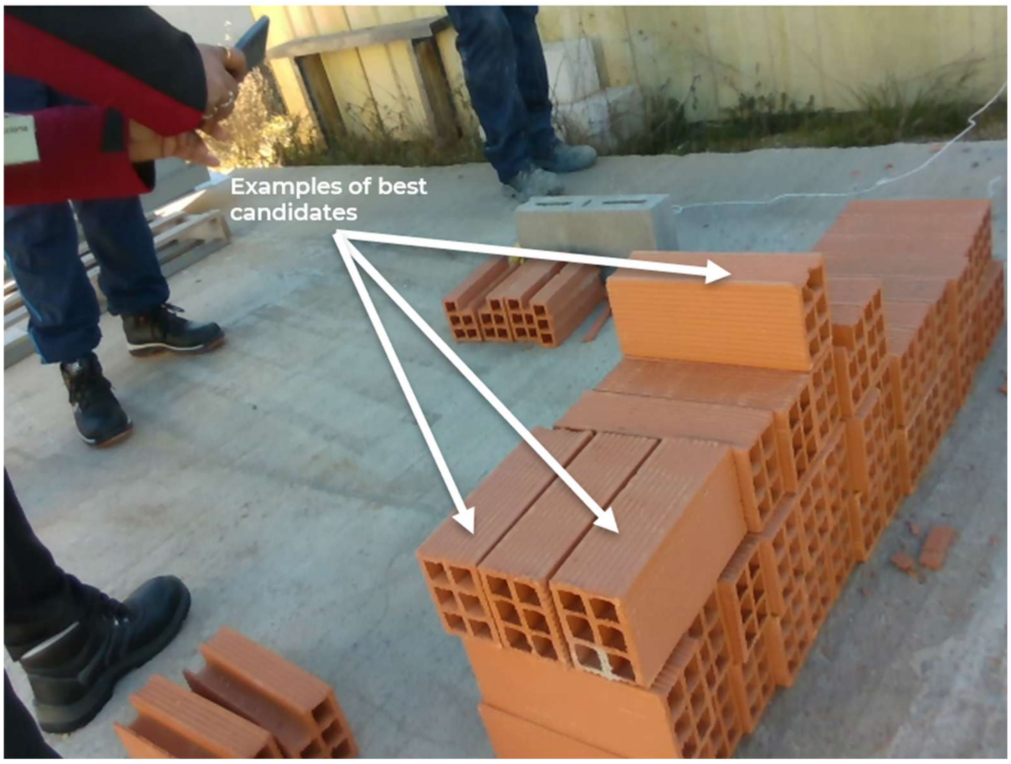


Figure 17 – Example of best candidate bricks for grasping.

The pipeline will be optimized and deployed on the Baubot robot platform (Section 1.1 & Figure 2). The best candidate selection is optimized and bundled together with the pose estimation algorithm. Furthermore, this module is developed in collaboration with *SINTEF* to test it with the robotic arm. The RGB image is recorded by the **Intel® RealSense™ D435i** [29] camera mounted on the robotic arm of the Baubot robot. The Intel® RealSense™ D435i as shown in Figure 18, contains an RGB camera sensor, stereo depth setup and an *inertial measurement unit* (IMU) sensor all integrated in a small form factor. With an Intel module and vision processor, the D435i is a powerful complete package which can be paired with customizable software. It's particularly known for its application in robotics and drones for navigation, environmental understanding, and object recognition. This makes using the D435i a good fit for our use cases.

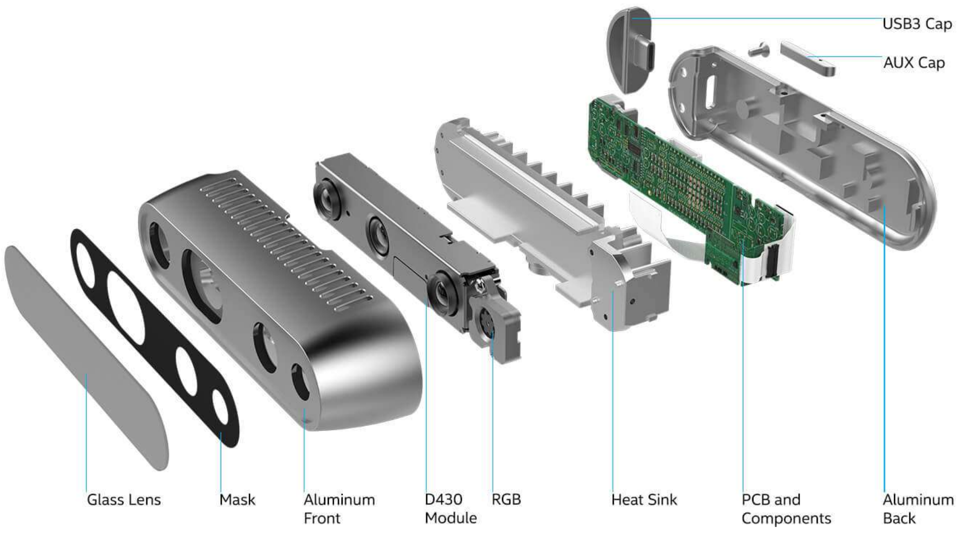


Figure 18 – Intel® RealSense™ D435i [29]

3.4. Evaluation metrics and results

The performance of deep learning algorithms is measured using some evaluation metrics. Evaluation metrics are used to evaluate deep learning models to provide a quantifiable measure of the model's performance. By comparing a model's predictions against actual outcomes, metrics give insights into the model's accuracy, efficiency, and areas of improvement. Overall, evaluation metrics offer an objective standard for assessing a model's efficacy and ensuring it meets the desired goals.

3.4.1. Evaluation of 2D detector

The performance of object detection models is usually evaluated using a combination of metrics that measure both classification accuracy and localization precision. As quality of 2D detections is also especially crucial, the YOLOv8 detector should be extensively evaluated. Here are some of the widely used metrics for evaluating 2D object detection:

Precision: This metric indicates the proportion of positive identifications (detected objects) that were actually correct. It is calculated as:

Precision = True Positives (TP) + False Positives (FP) / True Positives (TP)

Recall (Sensitivity): Recall calculates the proportion of actual positives that were correctly identified.

Recall = True Positives (TP) + False Negatives (FN) / True Positives (TP)

Average Precision (AP): For object detection tasks, precision and recall are usually plotted together as a Precision-Recall curve for different objectness score thresholds. The area under this curve is the Average Precision. AP gives a single-number measure of a model's quality across different threshold values.

Mean Average Precision (mAP): It computes the average of APs for all classes. In some contexts, mAP might be computed at different Intersection over Union (IoU) thresholds.

Intersection over Union (IoU): This metric measures the overlap between the ground truth bounding box and the predicted bounding box. It's defined as:

IoU = Area of Overlap / Area of Union

High IoU values indicate that the predicted bounding box closely matches the ground truth.

For practical evaluations, the choice of metric often depends on the specific problem at hand. For instance, in safety-critical applications, a high recall might be preferred over high precision, as missing an object might have more severe consequences than a false positive detection. On the other hand, in scenarios where manual validation of detections is costly, a high precision might be more desirable to reduce false alarms. For our use case, i.e., robot grasping, **High Precision** indicates that the detected objects are genuinely present in the scene and are viable grasp targets. A high precision means the robot will not waste time or risk errors by attempting to grasp falsely detected objects. For tasks involving delicate items or where misgrasps are costly, maximizing precision can be essential. This is particularly relevant when handling fragile items where a misgrasp due to a false detection could be detrimental. **High Recall** implies that the robot successfully detects and localizes a vast majority of viable objects for grasping in the scene. It ensures minimal missed opportunities.

Model	Box				Mask			
	P	R	mAP50		P	R	mAP50	Speed
Large	0.96	0.865	0.929		0.948	0.822	0.881	16.6 ms
Medium	0.952	0.864	0.925		0.952	0.817	0.881	12.2 ms
Small	0.943	0.831	0.911		0.948	0.788	0.859	6.6 ms

Table 1 – Evaluation metrics of 2D detections.

Therefore, for our application having an extremely high precision with a high enough recall is particularly important. The results of the evaluation done on the brick object are shown in Table 1. The Model type *Large*, *Medium*, and *Small* represents different sized of the neural network. The larger network has higher number of layers. But from the evaluations we can see all the variations perform well. The results also show that we achieve very high precision while also maintaining a high recall rate. In the table, **Box represents** the tight-fitting bounding box around the detected bricks and the **Mask** represents the semantic segmentation mask of the detected brick. The *mAP50* specifically means that detections are considered correct (a True Positive) if their IoU with the ground truth is greater than or equal to 0.50 (or 50%). In other words, for a detection to be deemed correct, it needs to overlap with the ground truth bounding box by at least 50%.

3.4.2. Evaluation of the Pose Estimator

BOP challenge [2][3] discussed in previous sections provides three pose-error functions by default, to evaluate the error of an estimated pose with respect to the ground-truth pose of an object. As the ZebraPose was evaluated as part of BOP challenge [2][3], these errors are directly computed first to get a baseline. These error functions are defined as

- ***VSD (Visible Surface Discrepancy)*** which treats indistinguishable poses as equivalent by considering only the visible object part.
- ***MSSD (Maximum Symmetry-Aware Surface Distance)*** which considers a set of pre-identified global object symmetries and measures the surface deviation in 3D.

– ***MSPD (Maximum Symmetry-Aware Projection Distance)*** which considers the object symmetries and measures the perceivable deviation.

An estimated pose is considered correct with respect to a pose-error function e, if $e < \theta_e$, where $e \in \{VSD, MSSD, MSPD\}$ and θ_e is the threshold of correctness. The fraction of annotated object instances for which a correct pose is estimated is referred to as **Recall**. The Average Recall (AR) with respect to a function e, denoted as AR_e, is defined as the average of the Recall rates calculated for multiple settings of the threshold θ_e and error specific tolerance values. The accuracy of a method on a dataset *D* is measured by $AR_D=(AR_{VSD}+AR_{MSSD}+AR_{MSPD})/3$, which is calculated over estimated poses of all objects.

The Average Recalls of all the three metrics for the brick object is shown in the below Table 2. These metrics are calculated using the default evaluation from BOP challenge. In particular, AR_{VSD} is the average of Recall rates calculated for τ ranging from 5% to 50% of the object diameter with a step of 5%, and for θ_{VSD} ranging from 0.05 to 0.5 with a step of 0.05. AR_{MSSD} is the average of Recall rates calculated for θ_{MSSD} ranging from 5% to 50% of the object diameter with a step of 5%. Finally, AR_{MSPD} is the average of Recall rates calculated for θ_{MSPD} ranging from *5r* to *50r* with a step of *5r*, where r = w/640 and *w* is the image width in pixels.

AR_D	AR_{VSD}	AR_{MSSD}	AR_{MSPD}
64.77	55.10	62.39	76.80

Table 2 – Recall rates based on the error function for 3D poses.

Although, we can see good Recall rates for the metrics, it does not reveal the whole picture. More particularly, it is important to understand which metrics affect the task of robotic grasping specifically. In Table 2, the recall values are computed and averaged over different thresholds as discussed before. Computing the values like this is more suitable for applications like AR, where the perceivable error is more important. These errors are then focused on the perspective projection i.e., the closer the objects are to the camera the more accurate they must be. In the task of robotic grasping/manipulation, where the robot arm operates in a 3D space, the **absolute error** of the estimated poses becomes extremely important. In the above defined three metrics, specifically, the MSSD is the metric which is more suitable for robotic grasping task. The other two functions, VSD and MSPD are more geared towards AR tasks.

Along with MSSD, evaluating Average Distance of Model Points proposed by Hinterstoisser et al. [29] which is geared more towards robotic grasping, will give a clearer picture on the performance of the ZebraPose. Instead of averaging the recall rates over all the different thresholds, for robotic grasping task, it would make more sense to observe the absolute error at these specific threshold values. Additionally, to handle the symmetric nature for some objects (like bricks), the Average Distance of Model Points metric also has a symmetric variant as well. The definitions of both are as follows.

ADD (Average Distance of Model points for objects) & ADD-S (for objects with Symmetry) [30]: Computes the average distance between 3D points of the ground truth object pose and the estimated object pose. For objects with significant symmetry, the ADD-S metric considers the closest point distances, which will be more appropriate.

The threshold used to calculate the metrics in Table 2 range from 5% to 50% percent of the diameter. The upper bound of 50% of the diameter is too high for the robotic grasping task. In line with the robot grasping scenario, instead of using percentages, the metrics are evaluated at specific values i.e., in terms of centimetres (cm). Therefore, while penalizing very small values and maintaining generalizability, the new θ_e selected for evaluating the metrics are: 1.5cm, 3cm, 4.5cm and 6cm. The results of this are documented in the below Table 3.

Error Metric (all bricks)	**$AR_{1.5cm}$**	**AR_{3cm}**	**$AR_{4.5cm}$**	**AR_{6cm}**	**AR_{mean}**
ADD-S	63.33	74.50	80.11	83.72	75.42
MSSD	22.15	44.65	56.55	64.85	47.10

Table 3 – ADD-S and MSSD metrics at specific thresholds.

The results show that the ADD-S metric performs well even with a very hard threshold of 1.5cm. The MSSD metric also performs well as it can recover almost 25 percent of poses. The important thing to note here is that these metrics are calculated for all the GT detections i.e., even for the bricks which are completely occluded and only one face is visible (as seen in Figure 9). To further analyse the metrics, only the GT where at least 85% of the brick is visible are considered. With this new constraint the metrics above are evaluated again. These results are documented in the Table 4. For the robot grasping task, the best candidate selection module (Section 3.3 &

Figure 16) will select the brick which also has very high visibility. So, evaluating only on bricks which have high visibility give a clearer picture. The results clearly show that they are significantly higher than the results evaluated for all GT. Especially, the ADD-S and MSSD metrics at 1.5cm which is a very hard threshold, also are much higher. The MSSD metric with 3cm threshold shows a drastic improvement over 1.5cm. The overall mean values are much higher when compared to the previous results in Table 4 as well. This shows that the designed pose estimation algorithm ZebraPose, is not only suitable for robot grasping task but it also predicts pose estimations with high accuracies.

Error Metric (85% visibility factor)	**$AR_{1.5cm}$**	**AR_{3cm}**	**$AR_{4.5cm}$**	**AR_{6cm}**	**AR_{mean}**
ADD-S	86.9	94.00	95.10	96.20	93.10
MSSD	30.15	61.20	80.32	87.45	64.9

Table 4 – ADD-S and MSSD metric evaluated for bricks with high visibility.

3.4.3. Qualitative results

The quantitative results in the previous section is computed on the synthetic data, due to the availability of GT information. However, in a real time use case GT information does not exist. Therefore, a qualitative analysis by using visualizations shows accuracy of the pose estimation model. In this section, the qualitative results on both synthetic and real images are shown.

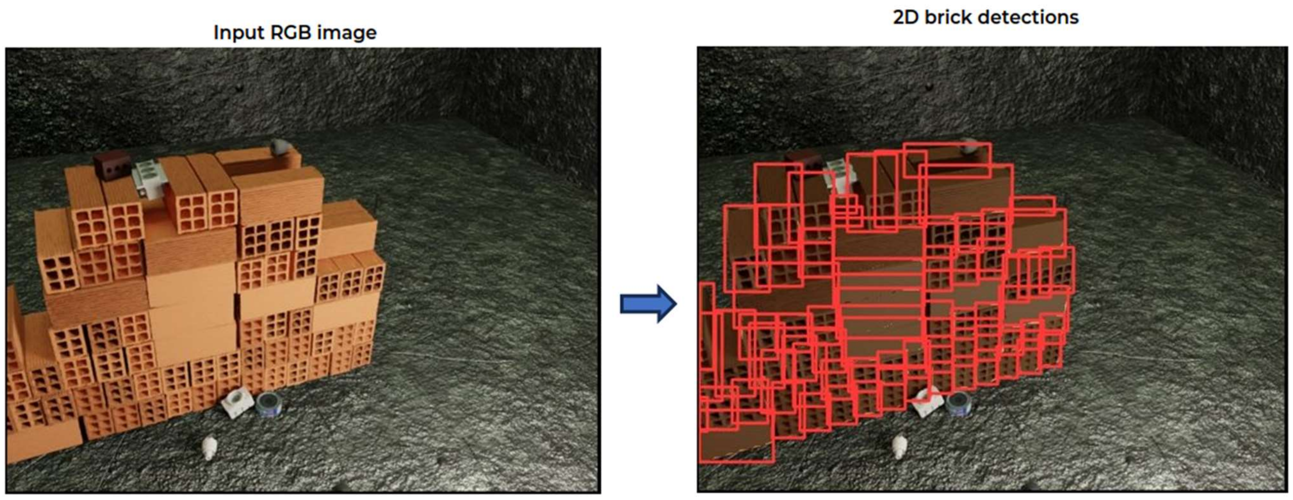


Figure 19 – 2D detection results.

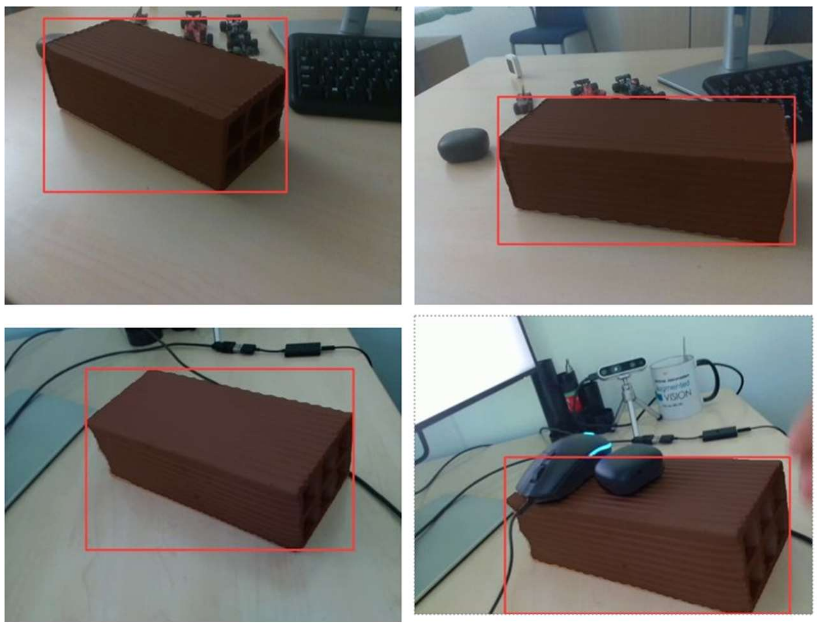


Figure 20 -2D detection on some real images.

The Figure 19 shows the visualization of the 2D detections and the segmentation masks on the synthetic dataset. It can be seen that, even the bricks at the bottom of the stack which are extremely occluded are also detected. Next, Figure 20 shows some images of the 2D detections and masks on some real images captured by the RealSense camera. For simplicity visualization with one brick is shown.

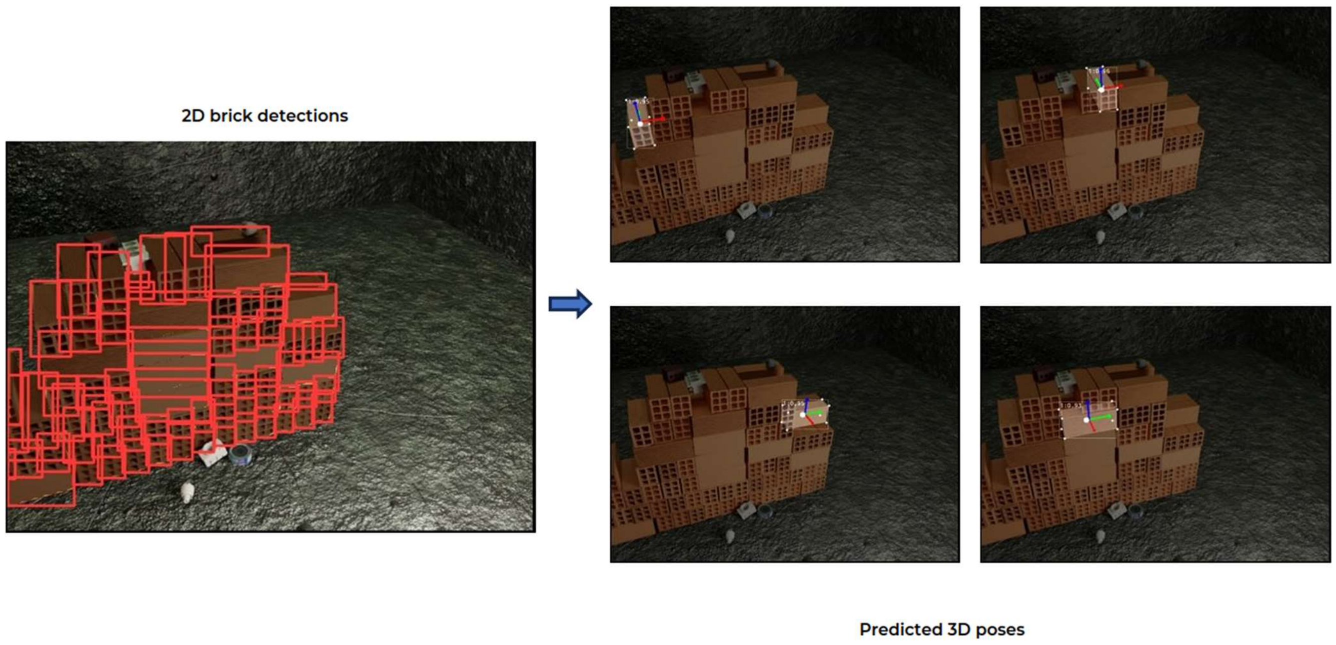


Figure 21 – Visualisation of 3D poses from 2D detections.

Subsequently, using the 2D detections the pose estimations are retrieved using the pose algorithm discussed in previous sections. The visualization of the pose i.e., the 3D model projected back to the image using the predicted pose is shown in Figure 21.

For the real time case, the visualizations are shown below in the Figure 22. The arrows at the centre of the brick depict the object coordinate axes and the white arrows depict the grasp area.

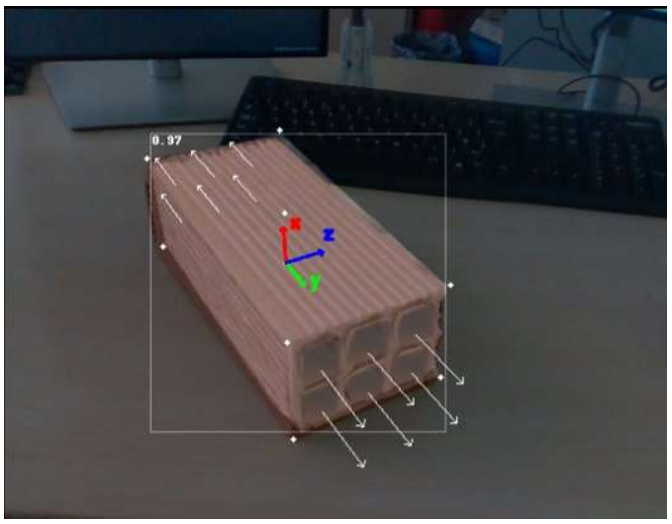


Figure 22 – Visualisation of 3D pose on a real image.

A video showing the visualisation of the realtime pose estimation using the RealSense camera is available at:

https://youtu.be/FnVuRsujjVE

4. Conclusion and next steps

Predicting 6DoF poses in the real world when there is no availability of datasets is a very challenging task. Towards this, the dataset generation pipeline and the designed algorithm tackle this problem. The dataset generation pipeline creates photorealistic samples of the real-world scenes by exploiting physically based rendering mechanisms of BlenderProc. As seen from the evaluation and results in the previous section, the presented approach for object detection and pose estimation are also able to generalize well in handling the domain gap from synthetic to real-world. Furthermore, the simple, efficient, and real time design of the end-to-end pipeline closes the loop, transferring research guided implementations to real-world practical scenarios. The ZebraPose algorithm of DFKI is not only state-of-the-art in accuracy but also is very efficient. Not only that, but it also runs at real time speeds, thanks to the simple binary encoding mechanism. Even with the overhead for a need of 2D detector the pipeline is designed to be still very fast. Even though, the accuracy of ZebraPose depends on the quality of the 2D detections, our modified YOLOv8 algorithm is very good at retrieving all the objects while also precisely localizing them in the image, as shown in the evaluation section.

As next steps, first, the designed pipeline is to be packaged into an easily installable software and integrated into the robotic platform (Figure 2). As it is impractical to install enormous number of hardware components on the robotic platform, the hardware availability and capacity will be limited.

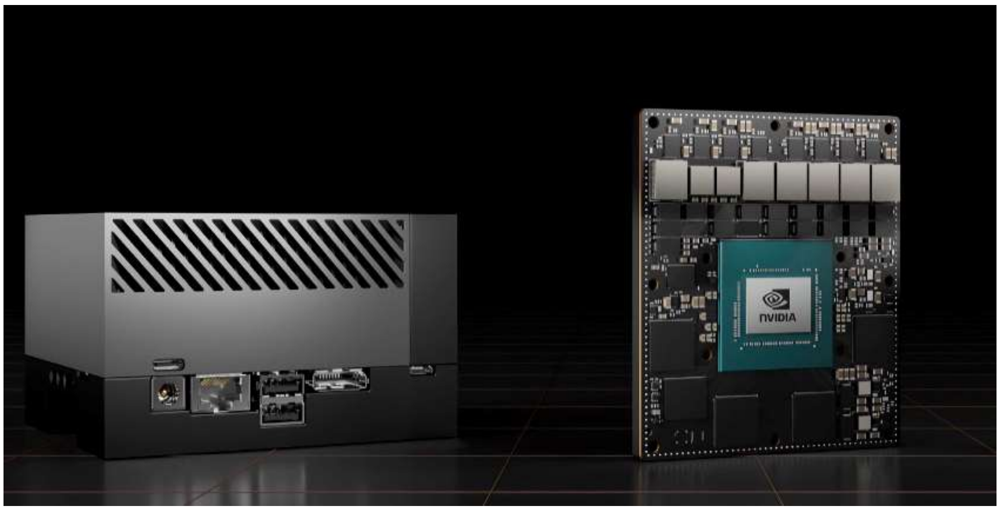


Figure 23 – Nvidia Jetson Hardware [31].

Therefore, the pipeline should also be able to run under computational restrictions. This means optimizing and finetuning the pipeline by using techniques like Quantization [32] in deep learning to ensure the neural network runs on lower hardware resources. Towards this, Nvidia Jetson [31] embedded computing boards are a good fit.

NVIDIA Jetson is a series of compact embedded computing platforms developed by NVIDIA, tailored for AI and deep learning applications. Each Jetson board features a potent GPU based on NVIDIA architectures. These boards come with integrated memory, essential I/O interfaces and most importantly with an accompanying JetPack SDK. Using the SDK developers get tools and libraries optimized for the platform, facilitating AI-focused projects, especially in robotics and edge devices, where the platform's power efficiency proves crucial.

Secondly, the pilot concept as part of *WP7-task 7.2*, involves Human Robot communication for the bricks grasping and handover task as show in the Figure 1 of the introduction section 1. Towards this, having the pose estimation setup locally available on the robotic platform, facilitates integration with other components. This is done in collaboration with other partners like *SINTEF, Baubot, RPTU* and *Acciona*. For example, once the hardware setup with the pipeline installed on it is deployed on the Baubot robot platform (Section 1.1 & Figure 2), our partner *SINTEF*, involved in the human-robot collaboration tasks (*T5.6*), will send an image or a sequence of images captured from the RealSense camera (Figure 18) mounted on the arm, to the deployed pipeline. In return, the pipeline will then send back the best candidate bricks in each image. More specifically, the 6DoF pose relative to the camera for each of the best selected candidate bricks will be sent back. The robotic arm with the gripper (Section 1.2 & Figure 3) will use this information to grasp the bricks and hand it over to the humans.

5. References

[1] T. Hoda\v{n} *et al.*, “BOP Challenge 2020 on 6D Object Localization,” *European Conference on Computer Vision Workshops (ECCVW)*, 2020.

[2] T. Hoda\v{n} *et al.*, “BOP: Benchmark for 6D Object Pose Estimation,” *European Conference on Computer Vision (ECCV)*, 2018.

[3] M. Sundermeyer *et al.*, “BOP Challenge 2022 on Detection, Segmentation and Pose Estimation of Specific Rigid Objects.” 2023. [Online]. Available: https://bop.felk.cvut.cz/home/

[4] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, “PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes,” in *Robotics: Science and Systems*, 2018. doi: 10.15607/RSS.2018.XIV.019.

[5] C. Wang *et al.*, “DenseFusion: 6D object pose estimation by iterative dense fusion,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019. doi: 10.1109/CVPR.2019.00346.

[6] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, “CosyPose: Consistent Multi-view Multi-object 6D Pose Estimation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020. doi: 10.1007/978-3-030-58520-4_34.

[7] M. Rad and V. Lepetit, “BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017. doi: 10.1109/ICCV.2017.413.

[8] O. Hosseini Jafari, S. K. Mustikovela, K. Pertsch, E. Brachmann, and C. Rother, “iPose: Instance-Aware 6D Pose Estimation of Partly Occluded Objects,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019. doi: 10.1007/978-3-030-20893-6_30.

[9] K. Park, T. Patten, and M. Vincze, “Pix2pose: Pixel-wise coordinate regression of objects for 6D pose estimation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019. doi: 10.1109/ICCV.2019.00776.

[10] L. Gi. Roberts, “Lawrence Roberts Machine Perception of Three Dimensional Solids,” 1965.

[11] D. G. Lowe, "Three-dimensional object recognition from single two-dimensional images," *Artif Intell*, vol. 31, no. 3, 1987, doi: 10.1016/0004-3702(87)90070-1.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int J Comput Vis*, vol. 60, no. 2, 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2006. doi: 10.1007/11744023_32.

[14] M. Calonder, V. Lepetit, M. Özuysal, T. Trzcinski, C. Strecha, and P. Fua, "BRIEF: Computing a local binary descriptor very fast," *IEEE Trans Pattern Anal Mach Intell*, vol. 34, no. 7, 2012, doi: 10.1109/TPAMI.2011.222.

[15] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *Int J Comput Vis*, vol. 81, no. 2, 2009, doi: 10.1007/s11263-008-0152-6.

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Commun ACM*, vol. 24, no. 6, 1981, doi: 10.1145/358669.358692.

[17] S. Hinterstoisser *et al.*, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011. doi: 10.1109/ICCV.2011.6126326.

[18] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017. doi: 10.1109/ICCV.2017.169.

[19] J. Rambach, C. Deng, A. Pagani, and D. Stricker, "Learning 6DoF Object Poses from Synthetic Single Channel Images," in *Adjunct Proceedings - 2018 IEEE International Symposium on Mixed and Augmented Reality, ISMAR-Adjunct 2018*, 2018. doi: 10.1109/ISMAR-Adjunct.2018.00058.

[20] G. Wang, F. Manhardt, F. Tombari, and X. Ji, "GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021. doi: 10.1109/CVPR46437.2021.01634.

[21] M. Sundermeyer, Z. C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3D orientation learning for 6D object detection from RGB images," in *Lecture Notes in*

Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018. doi: 10.1007/978-3-030-01231-1_43.

[22] S. Peng, X. Zhou, Y. Liu, H. Lin, Q. Huang, and H. Bao, “PVNet: Pixel-Wise Voting Network for 6DoF Object Pose Estimation,” *IEEE Trans Pattern Anal Mach Intell*, vol. 44, no. 6, 2022, doi: 10.1109/TPAMI.2020.3047388.

[23] Y. Su *et al.*, “ZebraPose: Coarse to Fine Surface Encoding for 6DoF Object Pose Estimation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2022. doi: 10.1109/CVPR52688.2022.00662.

[24] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM,” *IEEE Transactions on Robotics*, vol. 37, no. 6, 2021, doi: 10.1109/TRO.2021.3075644.

[25] G. Jocher, A. Chaurasia, and J. Qiu, “YOLO by Ultralytics.” Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[26] M. Denninger *et al.*, “BlenderProc,” Oct. 2019.

[27] T. Hodan *et al.*, “Photorealistic Image Synthesis for Object Instance Detection,” in *Proceedings - International Conference on Image Processing, ICIP*, 2019. doi: 10.1109/ICIP.2019.8803821.

[28] G. Pitteri, M. Ramamonjisoa, S. Ilic, and V. Lepetit, “On Object Symmetries and 6D Pose Estimation from Images,” in *Proceedings - 2019 International Conference on 3D Vision, 3DV 2019*, 2019. doi: 10.1109/3DV.2019.00073.

[29] Intel, “Depth camera d435i,” *Intel® RealSenseTM Depth and Tracking Cameras*. Jun. 2021. [Online]. Available: https://www.intelrealsense.com/depth-camera-d435i/

[30] S. Hinterstoisser *et al.*, “Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013. doi: 10.1007/978-3-642-37331-2_42.

[31] Nvidia, "Nvidia Jetson Orin," *NVIDIA*. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/

[32] PyTorch, "Quantization¶," *Quantization - PyTorch 2.1 documentation*. [Online]. Available: https://pytorch.org/docs/stable/quantization.html