IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 April 2023, accepted 10 June 2023, date of publication 22 June 2023, date of current version 30 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3288569

RESEARCH ARTICLE

Probabilistic Pose Estimation From Multiple Hypotheses

OMAR DEL-TEJO-CATALÁ[1], JOSE-LUIS GUARDIOLA[2], JAVIER PÉREZ[1], DAVID MILLÁN ESCRIVÁ[1], ALBERTO J. PEREZ[2], AND JUAN-CARLOS PEREZ-CORTES[2]

[1]Instituto Tecnológico de Informática, 46002 Valencia, Spain
[2]DISCA, Universitat Politècnica de València, 46022 Valencia, Spain

Corresponding author: Omar Del-Tejo-Catalá (odeltejo@iti.es)

This work is part of the FitOptiVis project [30] funded by the Electronic Components and Systems for European Leadership (ECSEL) Joint Undertaking under grant number H2020-ECSEL-2017-2-783162. Moreover, this work was partially funded by Generalitat Valenciana through IVACE (Valencian Institute of Business Competitiveness) distributed nominatively to Valencian technological innovation centers under project expedient IMAMCN/2021/1. It was also funded by the Cervera Network for R+D+I Leadership in Applied Artificial Intelligence (CEL.IA), co-funded by the Centre for Industrial and Technological Development, E.P.E. (CDTI) and by the European Union through the Next Generation EU Fund, within the Cervera Aids program for Technological Centres, with the expedient number CER-20211022.

ABSTRACT Pose estimation assesses the 6D pose of one or many objects in a scene. Considerable attention has been dedicated to the advancement of pose estimation algorithms capable of identifying the orientation of multiple objects within a scene in cases where partial occlusion occurs. However, only a few works focus on developing a parallelizable hypotheses-based estimator that naturally handles object symmetries. These algorithms should also tackle some issues: meaningless perspectives, objects with multiple uncertain local poses but a single global correct pose, and multiple correct poses. This paper proposes a novel probabilistic algorithm for pose estimation that addresses these issues. This probabilistic algorithm combines the information from multiple cameras to achieve a unique prediction that assembles global object information. The algorithm is tested over synthetic objects that simulate these issues. It achieves a rotation error below 1.5°, and a translation error of 1.5 pixels in the datasets used. Those results suggest that the algorithm can handle the mentioned issues up to a certain accuracy. Additionally, the method is compared against a state-of-the-art methodology of the LineMOD dataset. This comparison shows that our algorithm can compete against state-of-the-art algorithms in terms of accuracy.

INDEX TERMS Convolutional neural networks, deep learning, graph neural networks, pose estimation, orientation estimation.

I. INTRODUCTION

Pose estimation is a widespread research topic with many useful applications [1], [2], [3]. It is focused on assessing the 6D pose (rotation and translation) of one or many objects in a scene. Usually, pose estimation is performed using RGB or RGB-D images, or even complete 3D scenes, as input to some previously trained algorithm. However, the procedure is frequently shared for any kind of data. Mostly, some key reference points are extracted from the inputs based on previous knowledge about the scene, acquired during the training phase. Then, employing these reference points and their position, the 6D pose can be inferred.

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S. Raval.

As stated previously, pose estimation has several applications, ranging from industrial processes to Virtual Reality (VR) and Augmented Reality (AR) reconstructions, and human pose estimation [4], [5]. For instance, robotic arms get leverage from it to adjust themselves to improve their grip over the objects to pick up. VR and AR use pose estimation to enhance the experience and utility of simulations, such as the ones used to train pilots.

In particular, this paper aims to estimate the pose of free-falling objects captured simultaneously from several cameras. Such a system [6] allows quality inspection of any object without occluded areas, allowing a holistic quality assessment. The pose estimation process must have the ability to be performed concurrently to increase throughput. Thus, every camera must separately compute a hypothesis of the

pose. These hypotheses are refined in a central node to calculate the estimated pose of the object.

Moreover, each camera hypothesis should have an attached confidence value. Thus, the least confident hypothesis can weigh less in the combination or even be discarded in the final estimation refinement. This is critical to reducing the effect of possible symmetries and misleading predictions.

Not only does this paper consider possible symmetries within the shape of the assessed object but also in their texture. Many studies [3], [7], [8], [9], [10], [11], [12] address pose estimation from a texture-less object perspective. However, our approach must also consider texture alignment, which is somewhat included in [13]. For instance, from a holistic view, a die becomes asymmetric, and we can process even sphere-shaped objects if at least a key reference point is present and visible within the texture.

The datasets employed are made of 3D object views seen from different angles at a fixed distance captured simultaneously. Also, each simultaneous capture includes the object's pose at capturing time, i.e., the groundtruth of the dataset. Therefore, this paper hypothesizes that, given such datasets, we can build an algorithm that effectively predicts the pose of the 3D objects from all their views' estimations. For the hypotheses to be true, the algorithm must achieve an average estimation error of less than 5° for all test captures. This requirement is qualitatively and visually assessed, as the research team considers an error lower than 5° acceptable, and other authors [14], [15] report rotation errors above 10° for other pose estimation datasets. Moreover, an error lower than this is acceptable for the further quality assessments performed by the system described in [6]. Regarding translation, an error lower than 5 pixels is acceptable for the targeted use cases.

The contributions of this paper can be summarized as follows:

- Presenting an algorithm that achieves an acceptable error for the different datasets.
- To the best of our knowledge, describing a novel approach to pose estimation in which texture processing, multi-view inference, and symmetry robustness are key features of the algorithm proposed.
- Showing two different implementations of the algorithm, i.e., using Graph Neural Networks (GNNs) and Convolutional Neural Networks (CNNs), which achieve similar results.
- Comparing the results against a state-or-the-art algorithm for LineMOD dataset, a dataset to evaluate 6D pose estimation models.

In Section II, relevant topics related to the task and how other authors have addressed them are presented. Section III presents the methodology and datasets used in this work. In Section IV more detail about the implementation is given and the results are presented and discussed.

II. RELATED WORK

This section gathers some valuable concepts regarding pose estimation, divides them into different subsections, and describes how some authors have addressed them to give insight into the choices made in this work.

A. ROTATION ENCODING

There are many possible types of rotation encoding. The most common ones are quaternions, Euler angles, axis-angle, and rotation matrices. Usually, the decision is left to the authors whether to use one or another. However, [16] studies how each encoding affects the training phase of a neural network. It argues that rotation matrices are the only encoding that satisfies continuity and bijection. Quaternions and Euler angles either break the continuity constraint or more than one encoded rotation represents the same real rotation. For instance, one of the attributes of quaternions is that a quaternion q equals $-q$.

Even so, many works have used quaternions and Euler angles to train neural networks and achieved good results. For instance, [11], [14], [15], [17], [18] used Neural Networks along with quaternions or axis-angle encoding to estimate the pose of datasets such as (LineMOD, Occlusion, T-Less). Reference [8] did the same using Euler angles.

Depending on the encoding used, some issues should be addressed. For instance, quaternion and axis-angle encodings should have at least one of their four dimensions restricted to a hemisphere, thus allowing bijection at the expense of continuity. Euler angles and RPY (roll, pitch, yaw) encoding suffer from the Gimbal lock, which entails that, for some rotation values, one degree of freedom is lost. Rotation matrices are 9-dimensional, but one of the 3D vectors can be inferred using the other two thanks to the orthonormal property of the matrix, as pointed out in [16]. Thus, the 9D rotation matrix can be compressed to 6D without losing information. In [16], they further compressed the matrix to 5D but highlighted that this did not increase the accuracy of the models. Despite compressing the rotation matrix, it requires the highest amount of dimensions to encode the SO(4) rotations compared to 4D quaternions and 3D Euler angles. Nonetheless, as stated before, it has the benefit of continuity and bijection. Hence, it is the encoding used in this work.

B. CLASSIFICATION VS REGRESSION

Regarding the accuracy of the estimation, there is not a clear gap between algorithms trained for regression or classification. Usually, the authors' decision seems to be more technical rather than accuracy-oriented. For instance, [13] argues that classification is better than regression, whilst [8] argues otherwise.

The implementation in [15] allows hypotheses combination, which is the goal of our probabilistic perspective, using an algorithm trained with regression. Hypotheses combination will be further discussed in section II-D.

However, this regression approach is limited and doesn't fit the requirements of the probabilistic perspective we sought.

C. FEATURE EXTRACTORS

Traditionally, many authors [19], [20] have addressed the problem of pose estimation using feature extraction techniques such as Scale-Invariant Feature Transform (SIFT) or Speeded Up Robust Features (SURF). They extracted significant features from the objects under inspection and found correspondences between the original and captured models. Using these correspondences, they can estimate the pose. Nevertheless, these feature extractors cannot usually handle texture-less objects. Authors have recently turned their attention to Deep Learning approaches to solve this problem. Many possible structures have been used, such as single-shot detector architectures (EfficientNet [21], Single-Shot Detector (SSD) [22]), AutoEncoders, and plain CNNs.

The first ones [7], [8], [11], [13], [15], [23] have the advantage that they can handle multiple objects in a scene but strictly depend on a good detection phase that must be robust under occlusions.

The second approach, the AutoEncoder [12], seeks to compress and restore the input image to generate a meaningful reduced latent space. A codebook is generated from the reduced latent space of rotated images. In the testing phase, a test image is assigned the rotation of the train image with the closest latent space.

Lastly, CNNs [14], [17], [24], [25] take the images as input and estimate the rotation of the object under inspection. The way to compute the rotation varies among authors. Some [24] train the network to predict each pixel's coordinates and infer the rotation using them. Others [14] train the network to directly predict the rotation and ignore the translation component.

This work is grounded in the CNN approach and expands it to achieve a hypotheses-based algorithm.

D. SYMMETRIES AND UNCERTAINTIES

Due to random factors or the object's geometry, objects might have many possible associated poses. For instance, for every pose, a texture-less cube has 23 additional poses that look the same. Algorithms should be robust when this scenario arises. Some authors [8], [13] directly avoid this problem by eliminating possible symmetries from the training phase, or knowing a priori the relationships between possible symmetric poses and considering these within the algorithm's loss function. Others [15], [20] could solve this problem by allowing for noisy estimations using a hypotheses-related approach, especially from different perspectives of the object. Such is the case of this paper. The multiple hypotheses should agree, when the object is globally asymmetric, upon a single rotation, or cluster into the possible rotations if the object is not globally asymmetric like, for instance, the texture-less cube.

E. TEXTURES IN POSE ESTIMATION AND DATASETS

Effective and efficient solutions exist for the problem of pose estimation of textured objects without symmetries or occlusions. However, pose estimation in real environments is still often a challenging task. Thus, many researchers have been trying to face harder scenarios. For instance, when there are several objects in the scene, the objects under inspection have no clear reference keypoints, or the objects are symmetric. Thus, new datasets have been built to tackle more realistic cases. Such datasets include LineMOD, Occlusion, and T-Less.

Most of those datasets have more than one object in the scene. Also, the objects have no texture, so texture keypoints cannot be extracted. This work estimates the pose of a single object in the scene, independently of its geometry and without manually providing its symmetries to the algorithm. Therefore, the scenarios described in public datasets are beyond the scope of this paper, and suitable datasets had to be created. These datasets are described in Section III-A. Remark that this approach, despite not having complex escenarios, it is not abstract and is applicable for industrial inspection.

In summary, the focus of this paper is to design an algorithm that, unlike other authors [3], [20], [26] whose focus is on complex scenes with many objects and occlusions, can face any object shape seamlessly with no manual configuration. The object may be either textured or texture-less, with any kind of symmetry, i.e., a discrete symmetry along one axis, a continuous symmetry along one axis, or multiple discrete or continuous symmetries along more than one axis.

III. MATERIALS AND METHODS

First and foremost, suitable datasets must be created to address the problem. As the problem is quite specific, these datasets must be prepared to study the requirements imposed by the task. Commonly used datasets, such as LineMOD [1] and YCB-Video [11], are complex datasets as they present partially occluded objects in cluttered scenes. However, the objects themselves are not symmetric; therefore, these datasets do not allow us to assess the algorithm's shape robustness we seek. However, to compare the performance of our algorithm with other methods, we chose to include LineMOD's duck mesh.

A. DATASETS

Four synthetic datasets, seen in Figure 1, have been designed to exploit the algorithm's key features to be evaluated. These are:

- Sphere with a T. This dataset consists of a sphere with a ''T'' letter that marks the object's reference point. Most cameras capturing the object will not see the reference point. Thus, these cameras should give lower confidence in their predictions. When combining the predictions from multiple cameras, the ones that will contribute most must be the ones that captured the reference point.

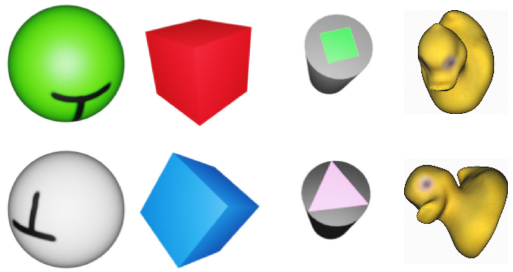


FIGURE 1. Synthetic dataset. From left to right: sphere with a T, cube, cylinder with a triangle and a square, and LineMODs duck.

- Cube. This dataset consists of a cube with no reference points; thus, any possible symmetric cube rotation is a valid pose.
- Cylinder. This dataset is made of cylindrical objects whose bases contain a triangle painted on one side and a square on the other. The square and the triangle are aligned to create the object's reference point, albeit no camera can see both simultaneously. Therefore, the pose of the cylinder can only be calculated by combining the predictions from the cameras that could see both polygons separately. On the one hand, the cameras that captured the triangle are expected to return a unique X-axis prediction and three equally likely Y-axis predictions. On the other hand, the cameras that captured the square should return a unique X-axis prediction and four equally likely Y-axis predictions. Then, after combining all the local predictions, a single Y-axis prediction should result.
- Duck. It has the property of not having symmetries, thus allowing us to compare our model with state-of-the-art algorithms from other datasets. In particular, this paper is compared against EfficientPose [8].

These datasets are generated using Blender 2.82. The scene contains the object randomly translated and rotated within a bounded working space and 14 cameras with perspective emulation enabled and equidistantly spaced over a sphere, whose normal optical vectors point towards the object. Diffuse background lighting was used to avoid shadow casting and reflections. Each simulated capture contains 14 512×512 RGB images (one for each camera) and a single groundtruth rotation and translation.

For each dataset, different object instances were created wherein changes in color and material were added to measure generalization in a synthetic environment. Each dataset has a single validation and test object variant. The number of training object variants varied for each dataset depending on the complexity of the shape. The cylinder and sphere datasets have three different object variants for training, whilst the cube and the duck datasets have just one. For each object variant, 6000 simulated captures were performed for training and 3000 for validation and testing. Therefore, there are 252000 training images for the cylinder and sphere datasets and 42000 for the cube dataset. Each dataset has 42000 validation and test images.

Data preprocessing should be performed for data reduction and to ease the training of the networks. In particular, we chose image background cropping. The square that contains the object whose center is the center of mass of the image is cropped from the image and resized to 128×128 pixels. u and v image coordinates and the scaling factor (the original size of the square divided by 128) are stored for each image. This information is then provided to the network during the training and inference.

The dataset was uploaded to Kaggle to allow experiment replication.[1]

B. POSE ESTIMATION NETWORK

This section presents the proposed algorithm structure and its training procedure. Figure 2 shows the different functional modules of the algorithm. These are:

- A feature extraction module where the images are processed and the pose-relevant features extracted.
- A perspective correction module to make the features perspective-independent (as if an orthographic camera captured the image).
- A rotation distribution predictor. There are two alternatives in this module: CNN and GNN.
- A translation prediction network that predicts an offset to the center of the segmented bounding box that estimates the camera projection of the object's centroid.

C. FEATURE EXTRACTION NETWORK

A VGG16 CNN [27], pre-trained with Imagenet [28] and cropped after the last pooling layer, was used to extract features from the input images. These layers were frozen until the first convergence of the algorithm. Two 3×3 convolutional and a global average pooling layer were appended to the end of the network to compress the features to 64 dimensions. The structure can be seen in Figure 7a.

D. PERSPECTIVE CORRECTION NETWORK

To some degree, every image captured by a perspective camera is affected by perspective distortion. The further the object is located from the camera center point, the more distorted it is. Hence, to increase precision, the extracted features should be transformed to correct perspective distortion.

A network consisting of two fully connected layers with 64 hidden dimensions takes the features and the position of the object's bounding box center in the image (u and v coordinates) and is trained to return the transformed 64-dimensional undistorted features. More detail can be seen in Figure 7b.

Once the features are extracted, and the perspective distortion is corrected, rotation and translation must be predicted. Regarding rotation, this paper addresses the problem in two alternative ways: using CNNs to predict two 2D maps that encode the rotation matrix and using GNNs to predict the probability of two sets of discrete points that encode the

[1] https://www.kaggle.com/itiresearch/pose-estimation

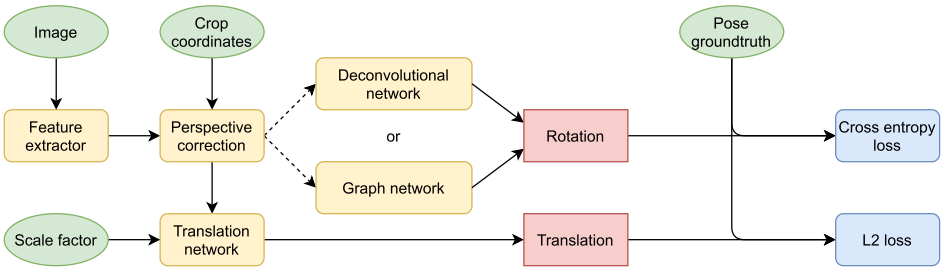


FIGURE 2. Algorithm's training pipeline.

rotation matrix. There is more detail on this in the following sections.

E. MAP ROTATION PREDICTION

As discussed above, we encode the rotation R of an object using the $\mathbb{R}^6$ compressed version of its rotation matrix. Unlike other authors who estimate the rotation directly, we train the algorithm to predict the probability density function of the axes R_x and R_y of R. In this implementation, each probability density function is encoded into two 2D maps M_x and M_y of BxB pixels. The pixels of these two maps correspond to the discretized spherical coordinates elevation ϕ and azimuth θ for each of the axes. This 2D representation has the drawback that the azimuth component θ is cyclic, i.e.:

$$\lim_{\theta \to 2\pi} M(\phi, \theta) \approx \lim_{\theta \to 0} M(\phi, \theta) \quad \forall \phi \in [0, \pi] \tag{1}$$

Both the rotation map encoding function and its inverse must allow for this.

To build the groundtruth maps, we transform each pixel in the map, which is related to some elevation ϕ and azimuth θ, to the 3D coordinate space of the R_i vectors. Then, given a Gaussian probability distribution $P \sim \mathcal{N}(R_i, 3e^{-2})$ centered in R_i, and with an acceptable standard deviation experimentally set, we compute the map M_i so that:

$$M_i(\phi, \theta) = P(\mathcal{S}(\phi, \theta)) \quad \forall \phi \in [0, \pi], \ \theta \in [0, 2\pi], \tag{2}$$

where $\mathcal{S}$ is the transformation function from spherical coordinates to 3D coordinates. Instead of the traditional Euclidean distance to compute the probability density function, this work uses the cosine distance, which is more meaningful for this spherical topology.

The goal is to develop a network $\mathcal{F}$ such that, given a set of input image features X, it returns a map $\hat{M}$ (the concatenation of $\hat{M}_x$ and $\hat{M}_y$) that predicts the rotation $\hat{R}$ minimizing the cross-entropy loss function $\mathcal{L}$:

$$\hat{M} = \mathcal{F}(X) \tag{3}$$

$$\mathcal{L}(M, \hat{M}) = -\sum_{\phi} \sum_{\theta} (M(\phi, \theta) \log \hat{M}(\phi, \theta) + (1 - M(\phi, \theta)) \log(1 - \hat{M}(\phi, \theta))) \tag{4}$$

This function is implemented with a Deconvolutional Neural Network. It takes the 64-dimensional features, projects them to a 16 × 16×16 tensor using a fully connected layer, and 4 deconvolution layers upsample the tensor to 256 × 256×2 dimensions. Therefore, for our experiments, B is set to 256. A softmax function is applied over each channel of the 256 × 256×2 tensor to restrict the area under the distribution to 1. An example of these maps can be seen in Figure 4 and Figure 5. These maps allow multiple local maxima, enabling the representation of several hypotheses at once. More information about this structure can be seen in Figure 7c.

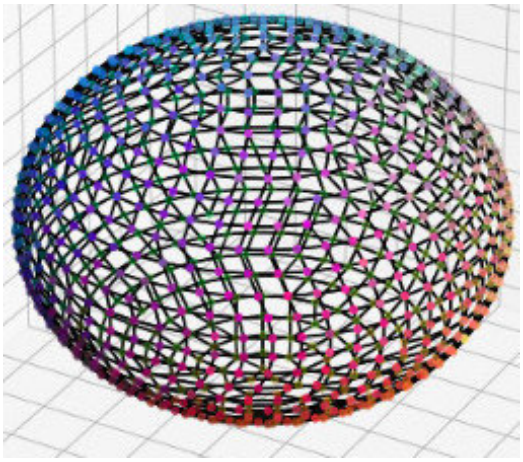


FIGURE 3. 3D undirected graph that represents the valid discretized rotations of R_i (which are points in a unitary sphere) connected with their 8 nearest neighbors to emulate the connections of a 3 × 3 image convolution.

F. GRAPH ROTATION PREDICTION

An alternative implementation is designed to predict a probability distribution over the real discretized $\mathbb{R}^3$ space of R_i. The goal is to substitute the convolutional layers with graph convolutions over the real topology. This modification allows for the cyclical structure of the topology and evenly distributes the sampling points over the unitary sphere of possible R_i. An example of the mesh of possible discretized rotations and their connections to perform the convolutions is shown in Figure 3.

2D maps and graph encoding are valid methods to compute the encoded rotation matrix, albeit using GNNs has additional benefits. Any 2D map conveys a projection from a 3D space into a plane. These projections are widely used, for instance, to create world maps. In particular, the one used

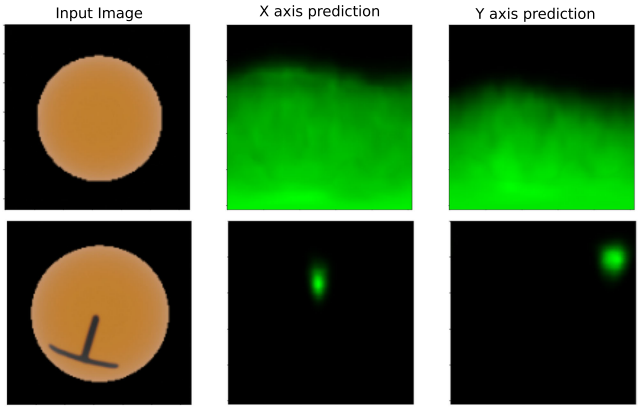


FIGURE 4. Rotation predicted by the algorithm for images with and without discriminative information. The input image is shown on the left, and the predictions for X and Y-axes are on the center and right, respectively. The upper region of the image without reference information has a low probability since an axis located in that region implies that the object's reference point would be visible. The maps are equalized for visualization purposes, so the brightness levels are not to scale.

in this paper is known as Mercator's projection. However, this projection does not maintain the relative distances among points. Therefore, the fact that discretization points are not evenly distributed results in varying precisions depending on the rotation. The proposed GNN solves this problem by keeping the topology of the 3D unitary sphere and performing the operations in that context.

This paper uses two different graph layers to create the rotation prediction network: graph convolutions and graph resampling layers. The first consists of a dense layer that projects the features of a given point in the graph into another dimensionality and combines the features from connected nodes using the mean. The second resamples the current topology to a given number of nodes. The new nodes check their closest neighboring nodes of the original graph and compute their features using an average weighted by distance.

The network takes the 64-dimensional features and projects them to 256 points with 16-dimensional features using a fully connected layer. Subsequently, these points are upsampled and the graph convolutions are applied. More information about the structure can be seen in Figure 7d.

G. TRANSLATION

The algorithm must also compute the translation of the object. Therefore, a fully connected network with three layers is designed to take the 64-dimensional uncorrected features and the scaling factor and predict a δu, δv, and depth. The scaling factor and the depth are highly correlated, but the image features are also crucial to predict the depth precisely. Given $u + \delta u$, $v + \delta v$, and the depth, we can compute the translation X, Y, and Z values.

H. MULTIPLE CAMERAS

So far, we have described the operations performed by a single camera. However, the complete method involves a combination of the information from several cameras to infer the real pose since we see in Figure 4 that a single camera cannot always assess the true rotation R of the object by itself. Furthermore, the combination can reduce the effect of noise, remove uncertain spikes in the distributions and strengthen the confidence around the true values of R_i. Thus, from now on, we will consider $\hat{M}^c$ as the hypothesis of the rotation R from camera c. We define the final prediction $\hat{M}$ of the distribution of R as

$$\hat{M} = \frac{1}{C} \sum_c \mathbb{X}_c(\hat{M}^c), \tag{5}$$

where $\mathbb{X}_c$ is the function that rotates the map or graph points using the extrinsic parameters of camera c.

$\hat{M}$ contains information about the discretized space coordinates $\mathcal{S}$ of both axes and their probability $\mathcal{P}$. Hence, the function $\mathbb{X}_c$ is defined as a rotation of the axis coordinates $\mathcal{S}$ using the camera extrinsic parameters $\mathcal{R}_c$ (only rotation is considered). Unfortunately, as the space is discretized, the original coordinates $\mathcal{S}$ should be kept in order to compare and blend with other cameras easily. Therefore, the new probability distribution of $\mathcal{S}$ can be estimated, e.g., as the weighted average of the 4 nearest points in $\mathcal{RS}$ rotated space.

Likewise, we can calculate the final predicted translation as the mean of the individual camera translations corrected with the known camera extrinsic parameters.

I. TESTING PHASE

During test and normal operation, $\hat{R}$ must be extracted from the predicted $\hat{M}$. To do so, the N maximum probabilities of each map $\hat{M}_x$ and $\hat{M}_y$ must be computed. The well-known Watershed algorithm has been used to extract conditionally independent maximums. Denoting by $\mathcal{W}$ the Watershed function, we apply:

$$\hat{R}_i = \mathcal{W}(\hat{M}_i) \quad \forall i \in [x, y] \tag{6}$$

This approach has a drawback. It suffers from a low-dimensional projection problem. The real space of R is $\mathbb{R}^6$ and we compressed it into two separate $\mathbb{R}^3$ predictions to reduce the spatial cost of discretizing a true $\mathbb{R}^6$ space. Therefore, noisy combinations can be drawn when extracting the maxima. Figure 5 shows a map $\hat{M}$ with 6 activated regions per axis. Each activation is related to a correct prediction of the space axes. However, not every combination is valid. For instance, the leftmost blob in the map $\hat{M}_x$ cannot be combined with the blob in the same position in the map $\hat{M}_y$. Both blobs are translated to the same 3D vector, so they are not orthogonal. Hence, after extracting all the possible combinations, the algorithm should discard the non-orthogonal ones.

IV. RESULTS AND DISCUSSION

This section contains the results of the approaches proposed on the datasets described in Section III-A along with some implementation details.

The algorithm was implemented using Tensorflow 2.4 [29]. We use the Adam optimizer with a learning rate of 7e-4 and an L2 regularization β of 1e-5. The training

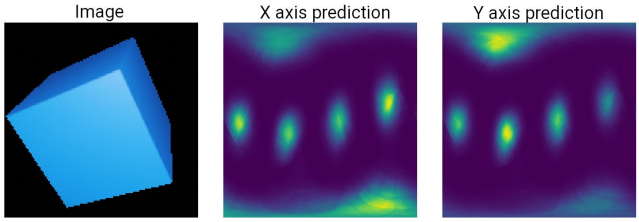


FIGURE 5. Example of a map $\hat{M}$ predicted by the network for a cubic object with no texture. Each map $\hat{M}_x$ and $\hat{M}_y$ has 6 activated regions that match the 6 possible configurations of each axis.

TABLE 1. Pose estimation error for each dataset's rotation and translation components.

Dataset | Errors	Deconv. rot. error	Graph rot. error	Translation error
Duck	$0.7° \pm 0.02°$	$0.8° \pm 0.02°$	$1.48px \pm 0.04px$
Cylinder	$1{,}679° \pm 0.07°$	$1.3° \pm 0.07°$	$1.74px \pm 0.01px$
Sphere with a T	$2.25° \pm 0.07°$	$1.5° \pm 0.05°$	$0.81px \pm 0.007px$
Cube	$0.924° \pm 0.02°$	$0.6° \pm 0.02°$	$1.27px \pm 0.01px$

phase encompassed 1000 epochs in an NVIDIA Tesla V100. Random color shifts and noise were included as data augmentation preprocesses.

A. METRICS

We evaluate the rotation error of the prediction using the geodesic distance metric described in [14]. The geodesic distance d_g is defined by the formula:

$$d_g(R, \hat{R}) = 2 \arccos \langle R, \hat{R} \rangle \tag{7}$$

The experiments for datasets with globally symmetric objects receive a particular treatment, for instance, the cube dataset. There are up to 24 valid rotations for this object to compute the rotation precision. Thus, two alternative methodologies are equally valid: either every single valid rotation is gathered and compared to the algorithm's prediction, or we calculate the 24 most likely predictions, compare them to a single groundtruth and take the one with the lowest error. In this paper, we used the latter.

Regarding the translation error, Euclidean distance was used. The rotation error was measured as an angle, and the translation error was measured in pixels at the nominal working distance of the camera using the intrinsic camera parameters. Pixels were selected to dissociate the system scale from the metric and be more comparable with other works.

B. EXPERIMENTS ON THE SYNTHETIC DATASET

The process was divided into two phases to compute the pose of the objects: the computation of meaningful internal representations of the images, and the estimation of the rotation, and the estimation of the translation.

Three separate training phases, one for each dataset, were first performed to minimize the rotation training loss described in (4). Transfer learning was used to train the initial steps until convergence. Then, a fine-tuning step was performed to update the first layers of the network with a learning rate of 1e-5.

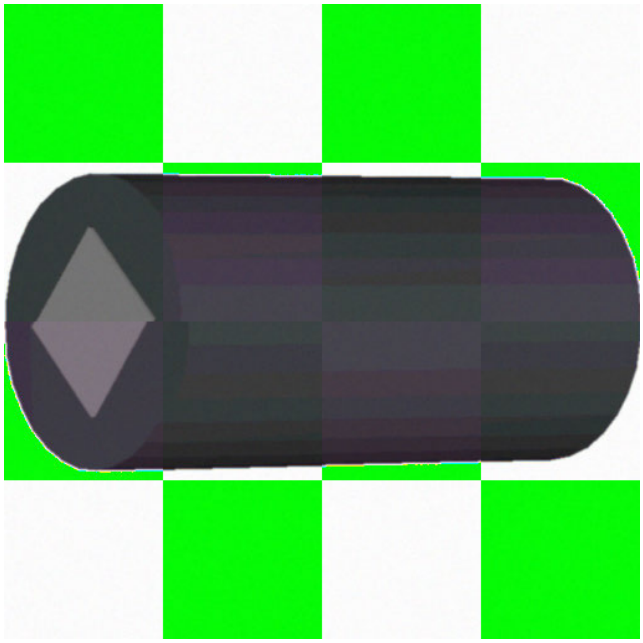


FIGURE 6. Difference of 1.5° between two different rotations. The image is divided in a checkerboard fashion, where a square belongs to one image and its neighbors to the other. A color shift was applied to one image to show where the image shifts occur.

Once the fine-tuning had converged, the translation network was trained using the internal representation of the images given by the rotation network. This procedure allowed a faster training phase for this network.

A comparison between the deconvolutional network and the graph network is performed, and the results for each dataset can be seen in Table 1. The graph network achieved lower error rates on the cylinder and the T-painted sphere but slightly higher on the cube dataset.

Regarding translation, an error of around 1.5 px was achieved for all the datasets, being the lowest 0.81 px for the T-painted sphere and 1.74 px for the cylinder. This difference in error is directly related to the geometry of the object. The cylinder has the most uneven geometry as, unlike the cube and the sphere, its points do not lie equidistant to the object's center of mass.

Although the comparison cannot be straightforward since the conditions, goals, and requirements are different, other authors report error rates on synthetic objects above 10° [14], [15]. As shown in Figure 6, a rotation error of 1.5° is perceptible but not substantial. The alignment can be followed by further comparisons between the captured and reference models, such as texture difference computation to compensate for possible anomalies. However, no texture anomalies were included during this work, so the algorithm's robustness against them has not been explicitly tested. Many data augmentation techniques can be added to ensure this generalization, but its effects on the accuracy were not measured.

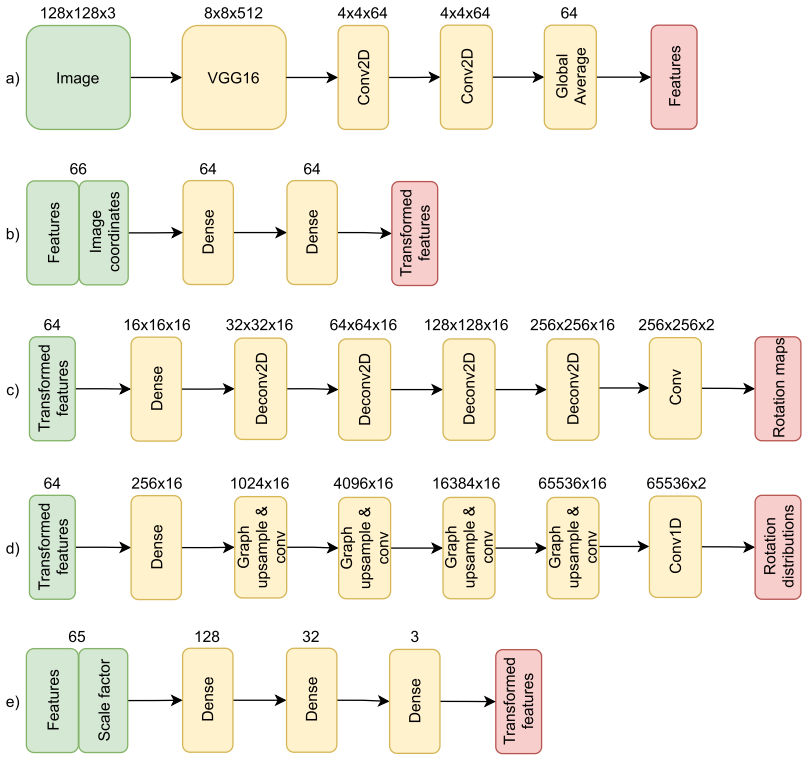


FIGURE 7. Structure of the networks used in the paper. A. Feature extractor, B. Perspective corrector, C. deconvolutional network, D. GNN, and E. translation network.

One of the strengths of this algorithm is that the input from each camera is not bound to a specific position, therefore allowing new cameras to be included in the system without building a new training set and conducting a new training phase. This feature is convenient for industrial environments as it can adapt to any configuration theoretically. Just by knowing the extrinsic parameters of the camera, as many cameras as needed can be easily added to the system.

Even so, during the training phase, one must ensure that the groundtruth rotations are adequately distributed. As it learns the probability distribution of a given view, this algorithm is quite sensitive to uneven groundtruth distributions. For rotations that are uncertain among three possible Y axis predictions, as in some views within the cylinder dataset, the algorithm might assign more confidence to one of them if there was a bias in the data set. We studied including a weight value that proportionally increases the loss of views whose groundtruth has not the highest confidence in the predicted probability distribution. This weight should become progressively more meaningful as the training converges. However, it did not significantly impact our training phase as the groundtruth was guaranteed to be equally distributed.

C. COMPARISON AGAINST EfficientPose

This section compares EfficientPose, currently the LineMOD dataset's state-of-the-art algorithm, with our methodology. For the following experiments we selected GNN approach as it outperformed the alternative deconvolution network. Both estimators are inherently different: one estimates the pose from a single image, whilst the other is designed to estimate it using several perspectives. However, this experiment is divided into several prediction metrics for a fair comparison. These metrics include:

- ADDs (original EfficientPose paper's metric for reference [8]). It measures the percentage of predictions that made the average closest distance between points in the meshes transformed with the predictions and the groundtruth below a 10% of the diameter of the mesh.
- Rotation error considering separate images, i.e., taking the likeliest prediction from our algorithm's predicted distribution and EfficientPose's single prediction. Some post-processes were added to both algorithm predictions to allow for object symmetries. Namely, each prediction was rotated 90° in each axis for the cube, and the one closest to the groundtruth was chosen. For the sphere and cylinder, predictions from perspectives that were not meaningful were filtered out, i.e., captures without the "T" for the sphere and captures that could not see the triangle or the square for the cylinder.
- Rotation error considering an image batch and combining predictions after the likeliest prediction from the distribution is computed, i.e., after our algorithm has already extracted the likeliest candidate and the respective post-processes stated above were applied.
- Rotation error considering an image batch and combining the probability distributions before extracting the likeliest prediction. This case is only applicable to our algorithm.

TABLE 2. Comparison between our algorithm and EfficientPose. Compares the likeliest prediction errors using separate images, the likeliest predictions mean errors at launch level, and the combination of the predicted distributions at launch level.

Object	Algorithm	ADDs per image	Rot. error per image	Rot. error per launch	Per launch combining maps
Duck	EfficientPose	0.999	$4.15^{o} \pm 0.1^{o}$	$1.63^{o} \pm 0.06^{o}$	-
	Ours	1.	$1.24^{o} \pm 0.009^{o}$	$0.306^{o} \pm 0.009^{o}$	$0.8^{o} \pm 0.02^{o}$
Cube	EfficientPose	0.878	$20.7^{o} \pm 0.2^{o}$	$5.09^{o} \pm 0.2^{o}$	-
	Ours	1.	$22.75^{o} \pm 0.2^{o}$	$7.58^{o} \pm 0.2^{o}$	$0.6^{o} \pm 0.02^{o}$
Sphere	EfficientPose	1.	$127.6^{o} \pm 1.^{o}$	$128.48^{o} \pm 3.^{o}$	-
	Ours	1.	$9.4^{o} \pm 0.6^{o}$	$5.61^{o} \pm 0.5^{o}$	$1.5^{o} \pm 0.05^{o}$
Cylinder	EfficientPose	0.956	$94.67^{o} \pm 1.^{o}$	$96.46^{o} \pm 4.^{o}$	-
	Ours	1.	$96.65^{o} \pm 1.^{o}$	$95.64^{o} \pm 4.^{o}$	$1.3^{o} \pm 0.07^{o}$

The results can be seen in Table 2. Here we can prove that our algorithm outperforms EfficientPose even in LineMOD objects. This finding does not imply that this methodology becomes state-of-the-art for the LineMOD dataset as it was not evaluated on LineMOD's test set because it contains single images with different backgrounds, which is out of the scope of the paper.

Furthermore, this experiment highlights that when the model's accuracy reaches a certain accuracy threshold, combining probability distributions may yield worse results compared to combining predicted rotations. However, in all other cases, combining the distributions instead of the predicted rotations resulted in lower estimation errors, indicating the advantages of delaying the extraction of the most likely prediction.

Moreover, the ADDs score demonstrates that the models were able to learn the estimation of the sphere's translation. However, EfficientPose struggled to predict the rotation, likely due to the absence of relevant pose information in the images. This uncertainty, where the same image corresponds to different targets, typically hinders automatic training procedures for models predicting single outputs. To overcome this limitation, manual tuning specific to the object being trained would be necessary to incorporate some prior knowledge. This justifies the inferior performance of EfficientPose on the sphere and the cylinder. However, in the case of the cube, both EfficientPose and our model considered all possible symmetric rotations and selected the one with the minimum loss, i.e., for experiments not utilizing map combination, we incorporated some prior knowledge about the object.

The cylinder object presents a similar scenario. It specifically evaluates the situation where multiple cameras have high confidence in a few potential solutions, and after merging all candidates, only a single solution remains. This is evident in the results presented in Table 2, where no model could provide valid solutions for the problem when combining predictions instead of probability maps.

Additionally, the results indicate that both models were capable of estimating most of the object poses in terms of ADDs, which focuses on the geometric properties rather than textures. This metric is widely used in the literature.

V. CONCLUSION

Pose estimation is a critical research topic with numerous applications in quality testing systems and other fields. However, developing a one-size-fits-all algorithm has been a significant challenge due to the difficulties posed by different object geometries. This work proposes a novel pose estimation algorithm that can effectively estimate the pose of objects with symmetric geometries using one or more cameras. While the accuracy of the algorithm is limited by the resolution of the camera and the discretization of the rotation space, it is still capable of estimating the pose of any object with a reasonable level of accuracy.

Although this methodology is more focused on automatically handling symmetries and combining different camera information, it performed better than EfficientPose on LineMOD's duck object in our case scenario.

Two different approaches to predict the rotations' distribution were compared, i.e., using graph convolutions or deconvolutional layers. Using graph convolutions proved to obtain better results on average than using deconvolutional layers. This is the case because graph convolutions adapt better to the output space's topology.

Our best approach achieves a rotation accuracy of below 1.5° and an average translation error of around 1.5 pixels, which is a promising result. However, further research is necessary to evaluate the robustness of the algorithm in real-world settings, including situations with slight texture changes or modifications in object size.

In summary, our algorithm presents a valuable contribution to the field of pose estimation, particularly in handling symmetric objects and combining information from multiple cameras. The results we have achieved so far are promising, and we believe that with further research and development, this approach could prove to be an essential tool for quality testing systems and other applications that require accurate pose estimation.

Among the possible limitations of this approach exists the fact that the space of rotation solutions is discretized. Although some interpolation is applied, in practice this usually implies a limitation in accuracy. Arguably, reasonable accuracy rates can be achieved with the resolution presented in this work. Moreover, another possible limitation is that the distribution of the training samples must be balanced. This means that, for every image that has multiple valid solutions, this solutions must be equally sampled during train so that no solution is preferred over another. Usually this can be easily handled as datasets are built synthetically. As future work, we plan to evaluate some techniques to address possible domain gaps between synthetic and real-world samples.

REFERENCES

[1] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes," in *Computer Vision—ACCV*, K. M. Lee, Y. Matsushita, J. M. Rehg, and Z. Hu, Eds. Berlin, Germany: Springer, 2013, pp. 548–562.

[2] J. M. Frahm, K. Köser, and R. Koch, "Pose estimation for multi-camera systems," in *Proc. Joint Pattern Recognit. Symp. (DAGM)*, in Lecture Notes in Computer Science, vol. 3175. Springer, 2004, pp. 286–293.

[3] T. Hodan, M. Sundermeyer, B. Drost, Y. Labbe, E. Brachmann, F. Michel, C. Rother, and J. Matas, "BOP challenge 2020 on 6D object localization," 2020, *arXiv:2009.07378*.

[4] R. Ranjan, V. M. Patel, and R. Chellappa, "HyperFace: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 1, pp. 121–135, Jan. 2019.

[5] Q. Dang, J. Yin, B. Wang, and W. Zheng, "Deep learning based 2D human pose estimation: A survey," *Tsinghua Sci. Technol.*, vol. 24, no. 6, pp. 663–676, Dec. 2021.

[6] J.-C. Perez-Cortes, A. Perez, S. Saez-Barona, J.-L. Guardiola, and I. Salvador, "A system for in-line 3D inspection without hidden surfaces," *Sensors*, vol. 18, no. 9, p. 2993, Sep. 2018.

[7] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," 2017, *arXiv:1711.08848*.

[8] Y. Bukschat and M. Vetter, "EfficientPose: An efficient, accurate and scalable end-to-end 6D multi object pose estimation approach," 2020, *arXiv:2011.04307*.

[9] T. Hodan, J. Matas, and Š. Obdržálek, "On evaluation of 6D object pose estimation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 9915. Springer, 2016, pp. 609–619.

[10] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T. K. Kim, J. Matas, and C. Rother, "BOP: Benchmark for 6D object pose estimation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 19–34.

[11] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arXiv:1711.00199*.

[12] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3D orientation learning for 6D object detection from RGB images," 2019, *arXiv:1902.01275*.

[13] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again," 2017, *arXiv:1711.10006*.

[14] S. Mahendran, H. Ali, and R. Vidal, "3D pose regression using convolutional neural networks," 2017, *arXiv:1708.05628*.

[15] F. Manhardt, D. M. Arroyo, C. Rupprecht, B. Busam, T. Birdal, N. Navab, and F. Tombari, "Explaining the ambiguity of object detection and 6D pose from visual data," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6840–6849.

[16] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," 2018, *arXiv:1812.07035*.

[17] F. Manhardt, W. Kehl, N. Navab, and F. Tombari, "Deep model-based 6d pose refinement in RGB," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 11218. Springer, 2018, pp. 833–849.

[18] J. L. Charco, B. X. Vintimilla, and A. D. Sappa, "Deep learning based camera pose estimation in multi-view environment," in *Proc. 14th Int. Conf. Signal Image Technol. Internet Based Syst. (SITIS)*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Jul. 2018, pp. 224–228.

[19] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, "3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints," *Int. J. Comput. Vis.*, vol. 66, pp. 231–259, Mar. 2006.

[20] A. Collet, M. Martinez, and S. S. Srinivasa, "The MOPED framework: Object recognition and pose estimation for manipulation," *Int. J. Robot. Res.*, vol. 30, no. 10, pp. 1284–1306, Sep. 2011.

[21] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, Jun. 2019, pp. 10691–10700.

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 9905. Springer, 2016, pp. 21–37.

[23] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4556–4565.

[24] K. Park, T. Patten, and M. Vincze, "Pix2Pose: Pixel-wise coordinate regression of objects for 6D pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 7667–7676, doi: 10.1109/ICCV.2019.00776.

[25] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "DeepIM: Deep iterative matching for 6D pose estimation," 2018, *arXiv:1804.00175*.

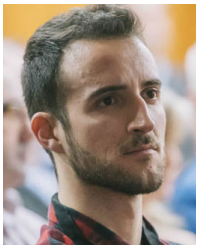
[26] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth," 2017, *arXiv:1703.10896*.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, *arXiv:1409.1556*.

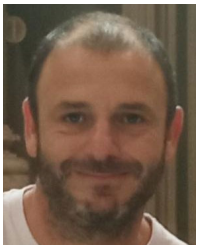
[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, Aug. 2009, pp. 248–255.

[29] M. Abadi, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 1–21.

[30] Z. Al-Ars, T. Basten, A. de Beer, M. Geilen, D. Goswami, P. Jääskeläinen, J. Kadlec, M. M. de Alejandro, F. Palumbo, G. Peeren, L. Pomante, F. van der Linden, J. Saarinen, T. Säntti, C. Sau, and M. K. Zedda, "The FitOptiVis ECSEL project: Highly efficient distributed embedded image/video processing in cyber-physical systems," in *Proc. 16th ACM Int. Conf. Comput. Frontiers*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 333–338, doi: 10.1145/3310273.3323437.



OMAR DEL-TEJO-CATALÁ was born in Valencia, Spain, in 1995. He received the engineering degree in computer science from the Polytechnic University of Valencia (UPV), and the master's degree in computer science. He is currently pursuing the Ph.D. degree with Instituto Tecnológico de Informática (ITI). He has been researching with the Pattern Recognition and Artificial Intelligence Group, ITI, since 2018. He is deeply keen on deep learning techniques applied to several fields, such as object detection, medical applications, reinforcement learning, and image classification.



JOSE-LUIS GUARDIOLA is currently an Industrial Engineer, a Computer Scientist, and a Bioelectronics Specialist with the Polytechnic University of Valencia. He has worked in 11 nationally funded projects, one European funded project with great impact, and several projects with private companies regarding signal processing, computer vision, and medical diagnosis. He has published six scientific papers in international congresses and two in Spanish congresses. His working experience is varied: computer vision and software engineering working with Inprobox Iberica SL, from 2003 to 2004, and Instituto de Investigación e Innovación en Bioingeniería, from 2005 to 2006. Since 2006, he has been with Instituto Tecnológico de Informática, and the Department of Pattern Recognition and Image Analysis, Polytechnic University of Valencia.

JAVIER PÉREZ is currently holds the master's degree in intelligence systems and the Ph.D. degree in computer science engineering with the University Jaume I (UJI), Castellón. Due to his research work, he has achieved nine publications in high impact journals, one book chapter, and more than 15 articles in important international conferences. Moreover, he is the coauthor in two European patents. Through his research career, he has worked in multiple European financed projects and Spanish government funded projects. He has more than six years of research experience in underwater robotics as a part of the Interactive and Robotic Systems Laboratory (IRSLab), UJI, and a Visitor with the Marine Australian Centre for Field Robotics (ACFR), The University of Sydney. In this period, he was in charge of developing computer vision, deep learning, 3D simulations, and benchmarking solutions for autonomous underwater robots. Since 2017, he has been a Software Engineer with the Research and Development Services Department, Technological Institute of Informática (ITI), developing 3D modeling and reconstruction techniques to enhance quality assurance processes. His current research interest includes artificial intelligence techniques applied to anomaly detection in manufactured products.

DAVID MILLÁN ESCRIVÁ received the degree in computer engineering and the master's degree in computer vision, artificial intelligence, and computer graphics from the Polytechnic University of Valencia (UPV). He has worked in some startups and companies, such as Skin Analytics Ltd., as the Computer Vision and Machine Learning Team Leader, and a Web Developer with Artres Comunicación. He was also worked in machine learning and computer vision fields for three years with the Emotion Research Laboratory. Since November 2018, he has been with Instituto Tecnológico de Informática (ITI) in computer vision, machine learning, and software engineering. He has published several books about OpenCV development.

ALBERTO J. PEREZ received the Ph.D. degree from Universidad Politécnica de Valencia, in 2005. He is currently a Lecturer with the Computer Science Department, Universidad Politécnica de Valencia, and collaborate with Instituto Tecnológico de Informática for more than ten years. He has published four articles in journals, seven in international congress, and two in national congress. He has been involved in more than ten public projects being the technical manager in two of them. His research interests include computer vision, features extraction or selection, and mobile robotics.

JUAN-CARLOS PEREZ-CORTES received the Ph.D. degree in computer science from the Polytechnic University of Valencia. He is currently a Full Professor with the Polytechnic University of Valencia. He is also the Director of the Pattern Recognition and Image Analysis (PRAIA) Research Group, Instituto Tecnológico de Informática (ITI). He has leaded and coordinated research projects funded by public national and international entities in medical imaging, industrial software, computer vision, pattern recognition, and free software. He teaches master's degree and Ph.D. courses in computer system's artificial vision and pattern recognition. He has published works in journals (15), books (3), books and conferences (17), and has been awarded by public and private entities.