Pattern Recognition 102 (2020) 107199

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

# Object recognition based on convex hull alignment

Robert Cupec*, Ivan Vidović, Damir Filko, Petra Đurović

*Faculty of Electrical Engineering, Computer Science and Information Technology Osijek, Josip Juraj Strossmayer University of Osijek, Kneza Trpimira 2b, Osijek HR – 31 000, Croatia*

ARTICLE INFO

*Article history:*
Received 25 January 2019
Revised 28 December 2019
Accepted 9 January 2020
Available online 13 January 2020

*Keywords:*
Object recognition
Shape instance detection
Depth image analysis
Convex hull
Shape alignment

ABSTRACT

A common approach to recognition of objects in cluttered scenes is to generate hypotheses about objects present in the scene by matching local descriptors of point features. These hypotheses are then evaluated by measuring how well they explain a particular part of the scene. In this paper, we investigate an alternative approach, which is based on alignment of convex hulls of segments detected in a depth image with convex hulls of target 3D object models or their parts. This alignment is performed using the Convex Template Instance descriptor. This descriptor was originally proposed for fruit recognition and classification of segmented objects. We have adapted this approach to recognize objects in complex scenes. Furthermore, we propose a novel three-level hypothesis evaluation strategy which can be used to achieve highly efficient object recognition. The proposed approach is evaluated by comparison with nine state-of-the-art approaches using three challenging benchmark datasets.



## 1. Introduction

The ability to recognize objects of interest is one of the basic functionalities of intelligent robots. The problem considered in this paper is how to detect instances of multiple *target objects* in a scene, given a depth image of this scene and a model database representing a set of 3D models of the target objects. This problem is referred to in this paper as *shape instance detection*. This is a fundamental problem in computer science with numerous possible applications. It has been investigated by many research teams, which have reported impressive results. Although various approaches to the problem discussed have been proposed, we believe that there are still ideas which are worth investigating in order to develop efficient and reliable solutions suitable for a variety of possible applications.

The basic principle of most of the existing approaches is a pipeline consisting of the following main steps: (i) computation of descriptors – vectors of values describing the shape, color or texture of the target objects, (ii) generating object hypotheses by matching the descriptors created from the scene to the descriptors created from the models of the target objects, and (iii) evaluating the generated hypotheses and selecting a consistent set of most probable hypotheses.

Descriptors used for object recognition can be classified into two broad classes: local descriptors, which describe part of an object surface, and global descriptors, which describe the whole object. Global approaches require objects to be segmented in a preprocessing step, which allows computation of a global descriptor [1,2] for each object. Since image segmentation into objects in cluttered scenes is a rather difficult task, global approaches are not suitable for object recognition in cluttered scenes. The LineMOD algorithm proposed by Hinterstoisser et al. [3] avoids image segmentation by using a template matching approach. Tejani et al. [4] integrate LineMOD into a random forest framework using a template-based splitting function.

Another problem with global approaches is how to deal with occlusion in cluttered scenes. Partial visibility of the target object can deteriorate the descriptor performance. Local approaches cope with this problem by generating multiple local descriptors for a particular object, where each descriptor describes a region of the object surface. Hence, even if the object is partially occluded in the scene, it can still be recognized, assuming that a sufficient number of distinguishable local descriptors are generated. Most local descriptors describe a neighborhood of a distinctive point selected on the object surface. This point is referred to as a *keypoint*. The keypoint neighborhood is commonly defined as a sphere of a predefined radius centered in the keypoint. If the sphere radius is too small, the content of the sphere can be insufficiently descriptive to allow efficient descriptor matching. On the other hand, if a too large radius is specified, the content of the sphere can include parts of adjacent objects in a cluttered scene.

* Corresponding author.
*E-mail addresses:* robert.cupec@ferit.hr (R. Cupec), ivan.vidovic@ferit.hr (I. Vidović), damir.filko@ferit.hr (D. Filko), petra.durovic@ferit.hr (P. Đurović).

https://doi.org/10.1016/j.patcog.2020.107199
0031-3203/

In this paper, we investigate an original approach, which differs significantly from the methods commonly used to solve the considered problem. Instead of detecting keypoints, we investigate an approach which is based on segmentation of the scene into convex segments. We assume that an object is represented in a depth image by a convex segment or a small group of adjacent convex segments. The convex hull of these segment groups is computed and described by the Convex Template Instance (CTI) descriptor [5]. The CTI descriptor is originally proposed as a global descriptor for fruit recognition. In [6], CTI descriptors are used to align the convex hulls of a query object with the convex hulls of the models in a model database for the purpose of object classification. A set of reference frames (RF) is computed for both the query object and each of the models. For each of these reference frames, a CTI descriptor is computed. Shape alignment is achieved by aligning reference frames corresponding to the most similar CTI descriptors. This approach is originally developed for shape alignment in an object classification framework, where a single isolated object is classified into one of several previously learned classes.

In this paper, we propose an adaptation of this approach for shape instance detection in complex scenes containing multiple objects, some of which can be partially occluded. Furthermore, we propose a simpler and more general method for defining local reference frames for CTI descriptors based on convex hulls.

Our main contributions in this paper are summarized as follows.

1 We propose a novel approach to generating object pose proposals based on alignment of convex hulls. The proposed method aligns convex hulls of convex segment clusters extracted from the input depth image with convex hulls of the target object models and their parts. An adaptation of the approach proposed in [6] is used to define a local reference frame (LRF) for the convex hull of the scene segment cluster and a set of LRFs for model convex hulls. The object pose proposals are generated by aligning these LRFs. The original approach [6] requires the input 3D mesh to be segmented into planar patches. For each planar patch a neighborhood is determined using a particular criterion, and a reference frame is defined from this planar patch and one of its neighbors. In this paper, we propose a new definition of reference frames, which does not require a 3D mesh segmented into planar patches as the input. Instead, it defines reference frames from convex hulls of an arbitrary 3D point set. This new definition is simpler and more generally applicable. Furthermore, we demonstrate by experiments that it provides sufficiently good object pose proposals for accurate shape instance detection. We named this approach CHAL (the acronym of Convex Hull ALignment).
2 We propose a novel highly efficient shape instance detection pipeline which uses the CHAL algorithm for hypothesis generation. The proposed method can be used to recognize objects in complex scenes with multiple objects, where some of the objects are occluded by others. The hypotheses generated by the CHAL algorithm are evaluated using criteria based on cues used in [7,8]. High computational efficiency is achieved by a three-level hypothesis evaluation and pruning strategy designed to achieve a balance between accuracy and computational efficiency. Each level rejects a significant number of hypotheses. More accurate but also more computationally demanding evaluation criteria are applied in each subsequent level. Consequently, a more demanding criterion is applied to a smaller number of hypotheses.

The proposed approach is evaluated using three challenging publicly available benchmark datasets. This experimental analysis demonstrates that the investigated approach can be used to achieve competitive performance in comparison to 9 state-of-the-art object recognition approaches.

## 2. Related research

Recognition of known rigid objects in 3D point clouds is today in a rather mature phase. State-of-the-art algorithms which solve this problem achieve high accuracy. A comprehensive survey of 3D object recognition methods is given in [9]. This section gives an overview of approaches which are most related to the approach proposed in this paper. The first part of this overview is structured according to the elements of the object recognition pipeline discussed in the Introduction, while the last section is dedicated to the approaches based on artificial neural networks.

### 2.1. Descriptors, keypoints and local reference frames

Local shape descriptors commonly describe the shape of an object surface in the neighborhood of a keypoint. The keypoint can be selected according to the distribution of the neighboring points [10], by identifying the local minima and maxima of shape indexes computed from principal curvatures [11], by Farthest Point Sampling [12], or simply by uniform sampling of a point cloud [8,13]. Some approaches describe the object surface by rotationally invariant descriptors. Ben-Yaacov et al. [14] proposed a rotationally invariant descriptor based on implicit polynomials, while Kuang et al. [15] used a histogram of the distances from the keypoint to the neighboring points and a distribution histogram of the heat diffusion function. Liu et al. [12] extracted features describing keypoint neighborhoods using a neural network. They cope with varying object orientations in the scene by training the neural network using point clouds synthetically generated by rotating the target objects. Many approaches achieve rotation invariance by defining a stable LRF in the keypoint and computing the descriptor with respect to this LRF.

Local shape descriptors used for object recognition are mostly based on oriented points. An *oriented point* is obtained by assigning to a 3D point a unit vector perpendicular to the local surface in the close neighborhood of that point. These vectors are referred to in this paper as *normals*. A simple descriptor computed from a pair of oriented points is proposed in [16]. The pair of points used to compute the descriptor is also used to define the associated LRF. A more informative description of the local shape can be achieved by descriptors computed from a local neighborhood of a keypoint. A popular descriptor – the Fast Point Feature Histogram (FPFH) [17] represents a histogram of features encoding relative positions of the points in a spherical neighborhood of the keypoint with respect to this keypoint and the orientations of the local surface normals. Another example of a histogram-based descriptor is the Signature of Histograms of OrienTations (SHOT) [18], which represents a histogram of normal orientations in the keypoint spherical neighborhood partitioned into several bins. The SHOT descriptor requires a stable LRF to be defined in the keypoint.

Zhong [19] proposes a method for assigning an LRF to a given keypoint based on the distribution of points in a spherical neighborhood of this keypoint. The axes of this LRF are parallel to the eigenvectors of the scatter matrix of the points in the neighborhood. Keypoints with stable LRFs are selected by requiring that the eigenvalues of the neighborhood scatter matrix are sufficiently different. A similar approach is applied in [1] for computing a global descriptor of the whole object, where the axes of the object reference frame are defined as eigenvectors obtained through principal component analysis of the point cloud of the target object. A method which can be used to disambiguate the signs of the axes obtained from eigenvectors is proposed in [20]. This method is also applied in the object recognition approach proposed in [21], where

the descriptor is built from a set of keypoints and each keypoint is assigned a LRF. Another approach to defining LRFs using point distribution in the sphere neighborhood of a keypoint is proposed in [22]. The keypoint normal is taken as the z-axis of the LRF and the x-axis is computed by aggregating the weighted projection vectors of the neighborhood points.

The approaches based on local point distribution require the local surface of a keypoint to contain sufficient information for defining three orthogonal axes unambiguously. Since such surfaces are not always available, descriptors which do not require a fully defined LRF have a clear advantage. The Spin Images approach [23] requires a keypoint and only one axis, i.e. the keypoint normal. The descriptor is computed for a given keypoint by rotating a plane around the keypoint normal and accumulating the neighborhood points with particular coordinates in that plane.

Mian et al. [24] computed the z-axis of LRFs as the average of the normals of two oriented keypoints and the x-axis by the cross product of these two normals. The drawback of the approaches using pairs of keypoints to define LRFs is that the number of point pairs increases with the square of the number of points. This number can be reduced by selecting the point pairs using some heuristics or by random sampling, as proposed in [24]. However, determining an appropriate number of pairs which must be randomly selected in order to ensure the correct correspondence between a scene point pair and a model point pair is an open problem.

The approach applied in this paper defines the axes of LRFs using the convex hull of a 3D point set. It does not require a stable keypoint. Instead, the translational component of the object pose proposal is determined by computing the optimal translation which minimizes the difference between the scene CTI descriptor and the model CTI descriptor.

### 2.2. Hypothesis generation, evaluation and selection

Object hypotheses are generated by matching descriptors computed from the scene to descriptors computed from the target object models. In the case of approaches which assign a LRF to every keypoint, a single match can be used to generate a hypothesis. Papazov and Burschka [7] generated hypotheses using RANSAC, where pairs of randomly sampled oriented points from the scene are matched with pairs of model points. The LRF defined by these two oriented points is used to estimate the object pose in the scene. Another approach is to generate hypotheses from groups of matches. The method proposed by Aldoma et al. [8] generates hypotheses by a correspondence grouping approach, which forms groups of correspondences satisfying a geometric consistency criterion. From these correspondence groups, hypotheses are generated using RANSAC to obtain initial poses.

Object recognition approaches based on local descriptor matching usually generate a large number of hypotheses. These hypotheses are then evaluated and a value describing the hypothesis plausibility is assigned to each of them. Evaluation of the generated hypotheses can be performed by transforming the model associated to a particular hypothesis with the transformation associated to this hypothesis and determining the number of scene points, which are sufficiently close to the transformed model points. This number is referred to in [7] as the hypothesis *support*. The second cue is penalization of visible model points which occlude visible scene points. These points are referred to in this paper as *transparent model points*. Since it is assumed that the objects of interest are not transparent, the points of the model projected onto the depth image of the scene should not occlude scene points visible in the image. These two cues are also used by our approach presented in this paper. In addition to these two cues, penalization of assignment of the same scene point to multiple hypotheses and hypotheses which assign multiple objects to the same smooth surface in the scene are proposed in [8].

In applications where an object recognition system must distinguish between multiple objects of very similar shape, the target object models must be precisely aligned with the corresponding subset of the scene points in order to correctly assess the similarity between different models and the object on the scene. For that purpose, the Iterative Closest Point (ICP) algorithm [25] is usually applied. This iterative procedure contributes significantly to the algorithm execution time. In order to improve computational efficiency, Wang et al. [26] proposed a fitting algorithm based on neural networks.

Aldoma et al. [8] proposed to use the assumption about the presence of planar surfaces on the scene. The presence of a dominant planar surface is exploited by detecting 3D clusters lying on that surface. Furthermore, the presence of planar surfaces is also exploited as an additional cue in hypothesis evaluation by penalizing hypotheses of objects intersecting a planar surface.

The final step of an object recognition method is a selection of a consistent hypothesis set. Papazov and Burschka [7] filtered the hypothesis set by removing a hypothesis if it explains the scene points which are also explained by a better hypothesis. Aldoma et al. [8] proposed an approach which generates combinations of hypotheses and evaluates each combination using an appropriate cost function. A combination with the minimum cost is selected as the final solution. The search for the minimum cost hypothesis set is performed by a meta-heuristic search method. Three techniques are considered as candidates: simulated annealing, local search and Tabu search.

The approach presented in this paper uses the CHAL algorithm for hypothesis generation. Accurate alignment of models with the scene, required for distinguishing between similar target objects, is achieved by an efficient implementation of the ICP algorithm. The final hypothesis set is obtained by a greedy method analogous to that applied in [7]. Although much simpler than the method used in [8], the experiments reported in this paper demonstrate that this method is sufficiently effective for moderately complex scenes.

### 2.3. Approaches based on neural networks

Recently, artificial neural networks have emerged as the method of choice in various computer vision problems. Neural networks are mostly used to detect or classify objects in RGB images. However, there are architectures which classify objects represented by 3D CAD models or 3D point clouds. A commonly used approach is to convert a 3D triangular mesh or a 3D point cloud into a voxel grid and apply a convolutional neural network to classify a target object into one of previously learned object classes [27,28]. In contrast to the shape instance detection problem considered in this paper, where an exact target object model is available, the object classification problem is to classify a previously unseen query object.

Another related problem, which is commonly solved by neural networks, is semantic scene segmentation, where the task is to assign each image pixel a label representing a semantic class of the object the pixel belongs to. An approach which uses a convolutional neural network to segment RGB-D images[1] into several categories typical of indoor scenes is presented in [29].

A shape instance detection approach based on neural networks is proposed by Kehl et al. [30]. Their approach detects a target object in RGB images. The neural network is trained using synthetically generated views of a colored target object 3D model. The applied neural network is based on the SSD architecture [31] and it

[1] RGB images with depth information assigned to image pixels

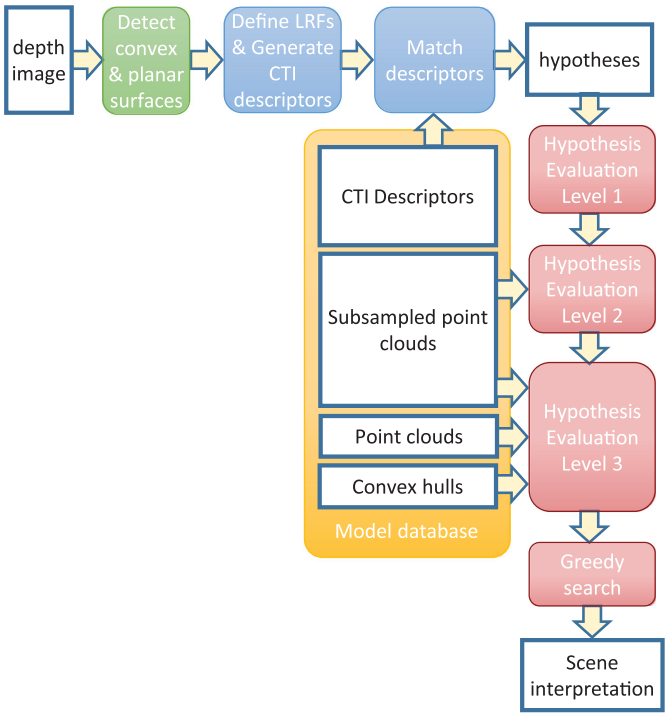


**Fig. 1.** Object recognition pipeline based on matching convex shapes. The pipeline consists of three main steps: detection of convex and planar surfaces (green), hypothesis generation (blue) and hypothesis evaluation and pruning (red) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

provides 2D detections in form of bounding boxes, where each box is provided with a pool of the most likely 6DoF poses for that instance. In a final step, each pose in every pool is refined, verified and the best pose is selected. The method can be applied to both RGB and RGB-D images. In the latter case, the final pose refinement step is conducted using depth information.

## 3. Object recognition pipeline

The object recognition approach proposed in this paper represents a pipeline whose input is a depth image of a scene and its output is a scene interpretation in the form of a list of hypotheses about instances of target object models appearing on the scene. The set of 3D models of all target objects is referred to in this paper as the *model database* denoted by M. The term *model* is used in this paper to denote the models of the target objects contained in M. Four representations of models are required by the proposed approach: (i) a set of oriented 3D points, (ii) a subsampled point set, (iii) a set of CTI descriptors with assigned LRFs, and (iv) the convex hull of the model. All these four representations are generated from a 3D model provided in the form of a dense triangular mesh, which is the basic model. The triangular mesh representing an object in the model database is denoted in this paper by $\mathcal{M}_j$, where $j$ is the model index. The same symbol is used to denote the set of vertices of the triangular mesh with assigned normals. Every model $\mathcal{M}_j \in$ M is assigned a RF denoted by $M_j$.

Analogously, the scene is represented by a dense triangular mesh, which is created from the input depth image. A triangular mesh can be obtained from a depth image very efficiently, e.g. by using the method proposed in [32].

The discussed object recognition pipeline is presented in Fig. 1. It generates a set of hypotheses about objects of interest being present in the input depth image, evaluates these hypotheses and selects a consistent subset of the most probable hypotheses as the final result.

As a preprocessing step, the scene and the models are segmented into approximately planar patches. For that purpose, the method applied in [33] is used. The points in which three adjacent planar patches meet are referred to in this paper as *vertices*. The vertices play an important role in the approach proposed in this paper, because the methods for detection of planar and convex surfaces and computation of convex hulls rely on these vertices. Since the number of vertices is significantly lower than the total number of points of the original dense mesh, the computational savings achieved by using the vertices instead of all points are significant.

The planar patches are aggregated into approximately convex surfaces using the method described in Section 4. The convex surfaces detected in the input depth image and the convex surfaces extracted from the models are referred to in this paper as scene segments and model segments, respectively. The hypotheses generated by the proposed approach associate scene segments to models $\mathcal{M}_j \in$ M. Since an object can be represented in the input depth image by multiple convex surfaces, scene segments are grouped into clusters. The term *scene segment cluster* (SSC) is used in this paper to denote a single scene segment or a group of adjacent scene segments. Scene segment grouping is described in Section 5.1.

The proposed approach estimates the object pose in the scene by aligning the convex hulls of SSCs with the convex hulls of models. In cluttered scenes, an object can be partially visible due to occlusion with other objects. In order to recognize partially visible objects, the convex hulls of SSCs are aligned not only with convex hulls of whole models, but also with convex hulls of model segments. In order to simplify writing, the term *model segment cluster* (MSC) is used to denote both model segments and whole models. Hypothesis generation by convex hull alignment is described in Section 5.

The generated hypotheses are evaluated by a three-level procedure described in Section 6. At each level, hypotheses are ranked according to a particular criterion. A limited number of the highest ranked hypotheses are passed to the next step, while the others are rejected. Each level is more accurate than the previous one, but also more computationally demanding. Hence, computational efficiency of the proposed approach is achieved by reducing the number of hypotheses at each step and applying more complex operations to a smaller number of hypotheses.

The proposed approach has several user-defined parameters. The values of these parameters, which are used in the experiments presented in Section 7, are also given in that section.

Now, let us introduce the notation which will be used in the remaining sections of this paper. A vector $v$ defined with respect to a RF $X$ is denoted by ${}^{X}v$. The left superscript index denoting a RF is used only when necessary. The pose of a RF $Y$ with respect to a RF $X$ is defined by a homogeneous transformation matrix ${}^{X}T_{Y}$. A homogeneous transformation matrix is composed of a rotation matrix ${}^{X}R_{Y}$ and a translation vector ${}^{X}t_{Y}$, where the relation between them can be described by:

$$ {}^{X}T_{Y} = \begin{bmatrix} {}^{X}R_{Y} & {}^{X}t_{Y} \\ 0 & 1 \end{bmatrix}. \tag{1} $$

## 4. Depth image segmentation

The first step of the proposed shape instance detection approach is to segment the input depth image into convex surfaces and large planar surfaces.

**Algorithm 1**
Mesh Segmentation into Convex Surfaces.

**Input:** $\mathcal{M}$, $\tau_{\text{minC}}$
**Output:**C = {$C_1$, $C_2$, ...}
1 : Segmentation of $\mathcal{M}$ into planar patches. The result is a set of planar patches F = {$F_1$, $F_2$, ...}
2 : B ← F
3 : $k \leftarrow 1$
4 : **Repeat**
5 : $F_i \leftarrow$ the largest planar patch in B.
6 : Remove $F_i$ from B.
7 : **If** $|F_i| < \tau_{\text{minC}}$ **then go to** line 28.
8 : $C_k \leftarrow F_i$
9 : $\tilde{n}_k \leftarrow n_i$
10 : $Q \leftarrow F_i$
11 : $V_{C_k} \leftarrow \emptyset$
12 : **Repeat**
13 : $F_m \leftarrow$ planar patch from $Q$ whose normal has the most similar orientation to $\tilde{n}_k$
14 : Remove $F_m$ from $Q$.
15 : **If** all vertices $p \in V_{F_m}$ satisfy (2) **then**
16 : Add $F_m$ to $C_k$.
17 : Add all vertices of $F_m$ to $V_{C_k}$.
18 : Remove $F_m$ from B.
19 : Update $\tilde{n}_k$ according to (6).
20 : Remove from $Q$ all planar patches $F_j$ which do not satisfy (3).
21 : Add to $Q$ all planar patches $F_j$ adjacent to $F_m$ which satisfy (3) and are not in $Q$ already.
22 : **end if**
23 : **until** $Q$ is empty.
24 : Add $C_k$ to C.
25 : Remove from B all planar patches contained in $C_k$.
26 : $k \leftarrow k$ x002B+ 1
27 : **until** B is empty.
28 : **return** C

### 4.1. Segmentation into convex surfaces

As explained in Section 3, the approach proposed in this paper is to segment the input mesh into approximately convex surfaces, represent these surfaces by the CTI descriptor and generate hypotheses by matching these descriptors to the descriptors of the same type representing target object models. In this section, the method used to segment the input mesh into approximately convex surfaces is described. Although several methods for segmentation into convex surfaces have been proposed [34,35], the method presented in this section has two properties, which are important for the specific application considered in this paper. First, it is highly computationally efficient. Second, it produces overlapping convex surfaces. This is important when dealing with target objects of non-convex shape which cannot be uniquely segmented into convex subsets. In these cases, the method described in this section produces multiple versions of segmentations into convex subsets.

The input to the algorithm is a triangular mesh $\mathcal{M}$ obtained from a 3D point cloud and the output is a set C of convex surfaces $C_i$. First, $\mathcal{M}$ is segmented into planar patches using the method described in [33]. This method is based on a region growing process which aggregates adjacent points of the input mesh into approximately planar sets. The boundaries between two adjacent planar patches is determined by searching for a path with minimum cost. This cost increases with the length of the path, where the path segment along the intersection line between the supporting planes of the two planar patches has zero cost. Thereby, planar patches with boundaries well aligned to the intersection lines between their supporting planes are obtained. Consequently, the obtained planar patch set can be regarded as an approximation of a polygonal representation of the input mesh. Each planar patch $F_i$ is assigned a set $V_{F_i}$ of vertices in which $F_i$ meets with two other adjacent planar patches.

Aggregation of planar patches into approximately convex surfaces is based on a region growing process explained by Algorithm 1.

The process is initialized by selecting the largest planar patch $F_i$, which represents the initial convex surface $C_k$. The process is initialized by a planar patch only if its size is greater than or equal to a threshold $\tau_{\text{minC}}$. This surface is then grown by adding adjacent planar patches $F_j$ which satisfy the convexity constraint with the convex surface $C_k$ grown so far. In addition to planar patches, a convex surface $C_k$ is assigned a set of vertices $V_{C_k}$. The convexity criterion a planar patch $F_j$ must satisfy in order to be joined to a convex surface $C_k$ is that all vertices $p \in V_{F_j}$ satisfy

$$n_l^T \cdot p - d_l \leq \varepsilon_{\text{convex}}, \quad \forall F_l \in C_k, \tag{2}$$

where $n_l$ is the normal of $F_l$, $d_l$ is the distance of the supporting plane of $F_l$ to the origin of the mesh RF, and $\varepsilon_{\text{convex}}$ is a tolerance set according to measurement noise. Furthermore, $F_j$ must satisfy

$$n_j^T \cdot p - d_j \leq \varepsilon_{\text{convex}}, \quad \forall p \in V_{C_k}. \tag{3}$$

If a planar patch $F_j$ satisfies the convexity criterion, it is added to the convex surface $C_k$ and all of its vertices are added to $V_{C_k}$. Tolerance $\varepsilon_{\text{convex}}$ is computed according to the planar patch uncertainty model adopted from [36]. The shape of a planar patch is approximated by an ellipse obtained by principal component analysis of its points. The ellipse lies in the supporting plane $\Pi_F$ of the planar patch $F$ centered in the centroid $\mu_F$ of the planar patch point set. The axes of the ellipse are aligned with the eigenvectors of the covariance matrix $\Sigma_F$ representing the distribution of the planar patch points in its supporting plane. The ellipse radii are twice the square root of the corresponding eigenvalues. For a given vertex $p$, tolerance $\varepsilon_{\text{convex}}$ in (2) and (3) is computed by

$$\varepsilon_{\text{convex}} = \sigma_{\text{noise}} \max\left(\frac{1}{2}\delta_{\text{M}}(\tilde{p}, \mu_F), 1\right), \tag{4}$$

where $\tilde{p}$ is the orthogonal projection of $p$ onto $\Pi_F$, $\sigma_{\text{noise}}$ is an experimentally determined measurement noise constant and $\delta_{\text{M}}$ denotes the Mahalanobis distance computed by

$$\delta_{\text{M}}(\tilde{p}, \mu_F) = \sqrt{(\tilde{p} - \mu_F)^T \Sigma_F^{-1} (\tilde{p} - \mu_F)}. \tag{5}$$

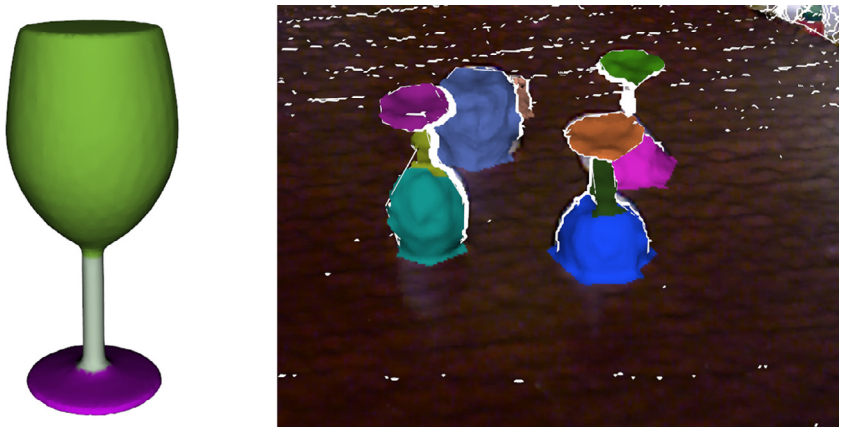

**Fig. 2.** Segmentation of a 3D object model (left) and a depth image (right) into approximately convex surfaces. Convex surfaces are painted in different colors.

If the orthogonal projection of $p$ falls inside the ellipse, then $\varepsilon_{\text{convex}} = \sigma_{\text{noise}}$. Otherwise, $\varepsilon_{\text{convex}}$ increases proportionally to the Mahalanobis distance $\delta_{\text{M}}(\tilde{p}, \mu_F)$.

A new planar patch considered for addition to a grown convex surface must be consistent with all planar patches already grouped into this convex surface according to the convexity criterion. Hence, the final shape of the convex surface is heavily dependent on the order in which planar patches are added thereto. The approach applied in our research is to select the next planar patch for inclusion in the grown convex surface according to the similarity of the orientation of its normal to the mean convex surface normal. The mean normal of a convex surface $C_k$ is defined by

$$\bar{n}_k = \frac{\sum_{F_i \in C_k} |F_i| \cdot n_i}{\left\| \sum_{F_i \in C_k} |F_i| \cdot n_i \right\|}, \tag{6}$$

where $|X|$ denotes the cardinality of a set $X$. In each iteration of the region growing process, the list $Q$ of candidates for inclusion in the grown convex surface $C_k$ is updated. At the beginning of each iteration, the candidate whose normal has the most similar orientation to $\bar{n}_k$is selected. The result of the described procedure is a set of convex surfaces referred to in this paper as *segments*.

As an example, the result obtained by applying the proposed mesh segmentation algorithm to a depth image and a 3D object model from the Kinect dataset [37] is shown in Fig. 2, where every segment is displayed in a different color.

### 4.2. Detection of large planar surfaces

Analogously to the approach presented in [8], we assume that the scene contains a dominant planar surface (DPS) on which the target objects are positioned. This situation is very common in robot manipulation tasks. Before segmentation into convex surfaces, detection of large planar surfaces is performed using a modification of Algorithm 1. The modification consists in the following:

1 The planar patch used to initialize a region growing process must have at least $\tau_{\text{minPl}}$ supporting points and the radius of its bounding sphere must be $\geq r_{\text{obj}}$. In the implementation of the proposed approach analyzed in the experiments reported in this paper, the bounding sphere is computed efficiently applying the method proposed in [38] to the planar patch vertices. The radius threshold $r_{\text{obj}}$ is determined according to the size of the largest object in the model database.
2 Instead of selecting the planar patch with the normal most similar to the mean convex surface normal in line 13, the largest planar patch in $Q$ is selected.
3 Instead of convexity conditions evaluated in lines 15, 20 and 21, a planarity condition is evaluated. A planar patch $F_i$ is considered to be coplanar with a planar surface $C_k$ if all vertices of both $F_i$ and $C_k$ lie on the least-squares plane of the union of the supporting points of $F_i$ and $C_k$ within a tolerance $\tau_{\text{PS}}$.
4 A planar patch is added to $Q$ only if it has more than $\tau_{\text{minPJ}}$ supporting points.

The DPS is the largest of large planar surfaces detected using the described approach. The planar patches belonging to large planar surfaces are not considered in detection of convex surfaces.

## 5. Hypothesis generation

In this section, a novel object hypotheses generation approach is proposed, which is based on aligning the convex hulls of segments detected in the input depth image with the convex hulls of models or their parts. The alignment of convex hulls is performed by aligning LRFs defined using the normals of the convex hull faces and CTI descriptors.

### 5.1. Local reference frames

In order to avoid generation of redundant LRFs, similarly oriented convex hull faces are clustered using a non-maximum suppression technique. Each cluster represents a subset of convex hull faces, which is assigned a normal. The normal assigned to a face cluster is computed as a weighted mean of the normals of the clustered faces, where each normal is weighted by the corresponding face area. Then, LRFs are defined from pairs of the obtained clusters. The non-maximum suppression technique used for face clustering is also used in subsequent stages of the object recognition approach proposed in this paper to avoid generating redundant object hypotheses. A general form of the applied non-maximum suppression technique is described by Algorithm 2. In the case of face clustering, the input set $X$ of the algorithm is the set of convex hull faces and the cost assigned to each face is the face area. A face is a neighbor of a face cluster if the angle between their normals is smaller than a predefined threshold $\theta_{\text{FC}}$. A face is merged with a face cluster by adding it to the cluster and re-computing the face cluster normal. The output set $Y$ of the algorithm is the set of face clusters.

In order to reduce the number of LRFs, only the face clusters whose total area is at least 4% of the largest face cluster area are considered when defining the LRFs. Pairs of face clusters with the angle between their normals $\geq 45°$ are formed. Each of these pairs is used to define an LRF. The normal of the first element of a pair

**Algorithm 2**
Non-maximum suppression.

**Input:**Set $X$
values $cost(x)$ assigned to every $x \in X$
neighborhood relation $neighbors(x, y)$ defined for all $x \in X$, $y \in Y$
**Output:**Set $Y$
1 : $Q \leftarrow$ list of elements $x \in X$ sorted according to $cost(x)$ in ascending order
2 : $Y \leftarrow \emptyset$
3 : $i \leftarrow 1$
4 : **While** $i <$ | $Q$ |
5 : $x \leftarrow i$th element of $Q$
6 : **For** every $y \in Y$
7 : **If** $neighbors(x, y)$ **then**
8 : Merge $x$ with $y$.
9 : **go to** line 13.
10 : **end if**
11 : **end for**
12 : Add $x$ to $Y$.
13 : $i \leftarrow i$ x002B+ 1
14 : **end while**
15 : **Return** $Y$.

represents the z-axis of an LRF and the x-axis of this LRF is defined by computing the cross product of the face cluster normals and normalizing the obtained vector to the unit length. The y-axis is uniquely defined by the x- and z-axis. The origin of the LRF is identical to the origin of the RF in which all vertices are represented. Nevertheless, the origin of the LRF is not required by the proposed CHAL algorithm.

In order to avoid redundant LRFs, the non-maximum technique based on Algorithm 2 is applied. The input to the algorithm is the set of face cluster pairs. The cost of each pair is the product of the total areas of the face clusters multiplied by −1. Two LRFs are neighbors if one of them can be obtained by rotating the other by the angle $< \theta_{\text{LRF}}$ about any axis. The merging step (line 8) is omitted. The result of the described procedure is a set of LRFs defined for a particular convex hull.

The described approach is applied to both the segments of models and the segments detected in the depth image. In the case of depth images, only the visible faces of convex hulls are considered. A face of a convex hull is considered visible if the face normal is oriented towards the camera.

In the case of convex objects, the segments detected using the method proposed in Section 4 represent whole objects. In general, an arbitrary object appearing on a scene can be represented by multiple convex segments. Hence, segments detected in the input depth image are grouped into clusters and a set of object hypotheses is generated for each segment cluster. A segment cluster is formed from one or multiple segments by examining all combinations of at least one and at most $n_{\text{SC}}$ segments and computing the minimum bounding sphere of the union of the considered segments. If the radius of this bounding sphere is $\leq r_{\text{obj}}$, then an SSC is formed from the considered segment combination. The convex hulls of the SSCs are computed and a LRF is defined for every SSC. The approach proposed in this section defines multiple LRFs for a particular convex hull. However, only a single LRF, the one generated from the face cluster pair with the lowest cost, is defined for every SSC. The LRFs defined for the SSCs are referred to in this paper as scene LRFs. In this paper, the scene LRFs are denoted by $L^{\text{S}}$ and their orientation with respect to the scene RF is defined by the rotation matrix ${}^{S}R_{L^{S}}$.

In the training phase, every model in the model database is segmented using the approach described in Section 4. The convex hull is computed for each model segment as well as for each model as a whole. A set of LRFs is defined for the convex hull of every MSC. These LRFs are referred to in this paper as model LRFs. The model LRFs are denoted by $L^{\text{M}}$ and their orientation with respect to the scene RF is defined by the rotation matrix ${}^{S}R_{L^{\text{M}}}$.

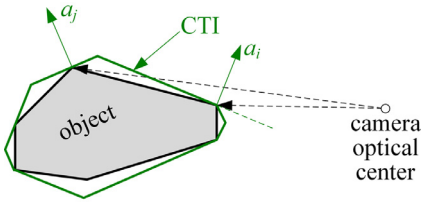


**Fig. 3.** A polyhedron (gray) and a CTI generated for this polyhedron (green). The $i$th face of the CTI is visible from the camera viewpoint, while the $j$th face is not (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

### 5.2. Convex hull alignment

The relative orientation of a model with respect to a SSC can be determined by aligning a scene LRF with a model LRF. The translation which aligns a model with a SSC can be determined using the method proposed in [6], which is based on CTI descriptors [5]. The idea of the CTI descriptor is to approximate a convex polyhedron by another convex polyhedron whose face normals belong to a finite set of unit vectors. Let $a_i$, $i$ x003D= 1, 2, ..., $n_{\text{CTI}}$ be a set of unit vectors. This vector set represents a *convex template* defining a set of all convex polyhedrons such that each face normal of each of these polyhedrons is parallel to one of the vectors $a_i$. A polyhedron belonging to this set is referred to as a *Convex Template Instance* (CTI).

Let $V$ be a set of vertices of an arbitrary convex polyhedron. The smallest CTI which contains $V$ is defined by

$$K(V) = \left\{ p \in \mathrm{R}^3 \middle| \max_i \left(a_i^T \cdot p - d_i\right) \leq 0 \right\}, \tag{7}$$

where

$$d_i = \max_{p \in V} \left(a_i^T \cdot p\right). \tag{8}$$

A CTI $K(V)$ is completely defined by the vector $d(V) = \left[\, d_1, \quad d_2, \quad \ldots \quad d_{n_{\text{CTI}}} \,\right]^T$, representing the CTI descriptor of the considered polyhedron. Each element of this vector represents the distance of the supporting plane of one face of the CTI from the origin of the RF in which the vertices in $V$ are defined. According to definition (8), the supporting plane of the $i$th face of a CTI computed from a vertex set $V$ is a tangential plane in one of the vertices $p \in V$ perpendicular to the vector $a_i$, as illustrated by Fig. 3.

A few examples of convex shapes and the corresponding CTIs are shown in Fig. 4. The presented CTIs are created for a convex template consisting of $n_{\text{CTI}} = 66$ unit vectors with approximately

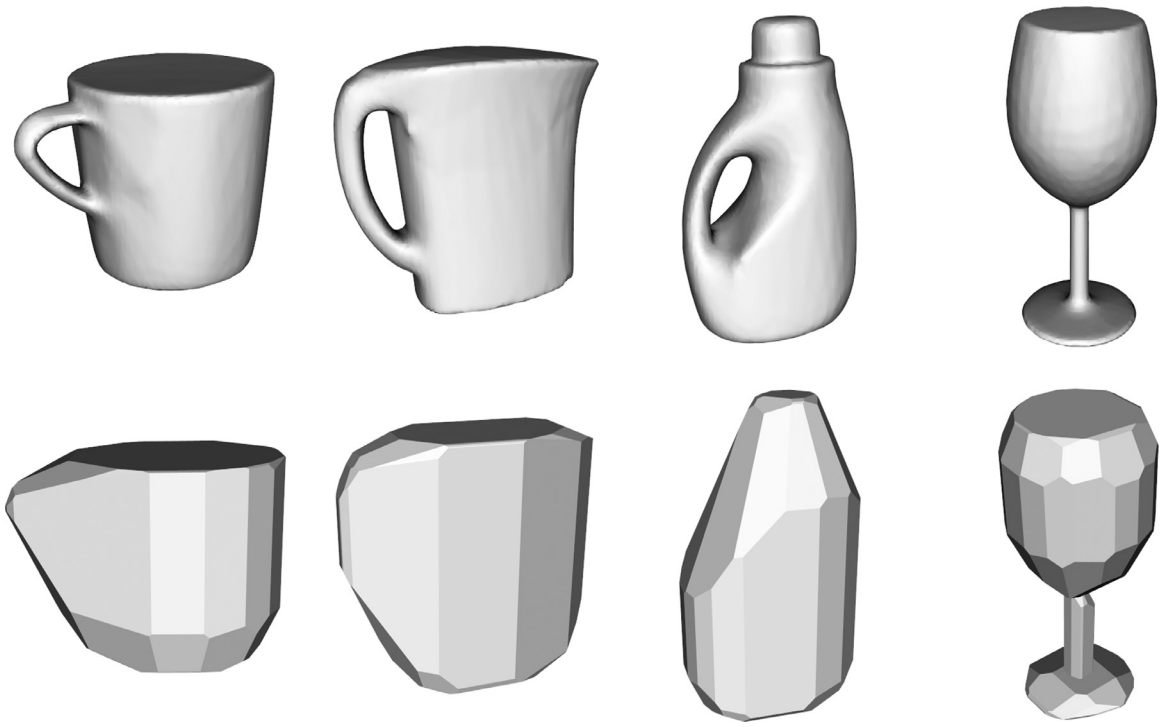

**Fig. 4.** 3D models (top row) and the corresponding CTIs (bottom row). The first three CTIs represent the whole shape of the first three objects, while the wine glass is represented by three CTIs representing its three convex segments.

uniform distribution on the unit sphere. This convex template is used in the experiments presented in Section 7.

A method for computing the optimal translation and scale, which aligns two polyhedrons using their CTI descriptors, is proposed in [6]. Scale invariance is an important property for the object classification problem addressed in [6], because query objects are supposed to have different sizes, and must be distinguished by their shape only. On the other hand, for the shape instance detection problem considered in this paper, the target objects have an exactly defined size, which can be used to distinguish one object from the others. In this paper, the approach proposed in [6] is adapted for the purpose of computing the optimal translation which aligns two polyhedrons.

The similarity between two convex polyhedrons can be measured by the Euclidean distance between their CTI descriptors. However, notice that a CTI descriptor defines not only the shape, but also the position of a polyhedron. If $V'$ is a vertex set obtained by translating a vertex set $V$ for a translation vector $t$, then

$$d(V') = d(V) + At, \tag{9}$$

where $A$ is the matrix whose rows are transposed unit vectors $a_i$. This can be easily shown from the definition of the CTI descriptor (8). The optimal translation which aligns two CTIs can be computed as the vector $t$ which minimizes $\|d' + At - d\|_2$. This is a least-squares minimization problem, whose solution is

$$t = \left(A^T A\right)^{-1} A^T \left(d' - d\right). \tag{10}$$

The approach based on the LRFs defined in Section 5.1 and the translation invariant CTI descriptor matching described in this section is referred to in this paper as Convex Hull Alignment (CHAL). The CHAL algorithm is described as follows.

Let ${}^{S}V_S$ be the set of vertices of a SSC defined with respect to the scene RF and let ${}^{M}V_M$ be the set of vertices of a MSC defined with respect to the model RF. The CHAL algorithm, presented as Algorithm 3, computes a set of $n_{MSC}$ alignment proposals between the considered MSC and SSC. Each of these proposals is defined by a transformation matrix ${}^{S}T_{M,i}$, $i$ x003D= 1, ..., $n_{MSC}$.

For the purpose of computational efficiency, the model LRFs and the belonging CTI descriptors can be precomputed off-line (lines 1, 2 and 7 of Algorithm 3).

### 5.3. Partial visibility

If an object is observed by a 3D camera from a single viewpoint, only a portion of its surface is visible in the acquired point cloud. Since every component of a CTI descriptor corresponds to a polyhedron face, only the components of the CTI descriptor corresponding to the faces whose normals are oriented towards the camera are defined. These components are referred to in this paper as *visible components* of a CTI descriptor. An example is shown in Fig. 3. A simple criterion can be used in order to determine whether a component of a CTI descriptor is visible or not. If a component of a CTI descriptor, defined with respect to the camera RF centered in the optical center, has a negative value, then this component is visible. The visible part $d^{\mathrm{v}}$ of a CTI descriptor $d$ can be extracted from $d$ by

$$d^{\mathrm{v}} = Zd, \tag{11}$$

where $Z$ is a *visibility matrix* with $n_{\mathrm{v}}$ rows and $n_{\mathrm{CTI}}$ columns, whose element $z_{ij} = 1$ if the $i$th element of $d^{\mathrm{v}}$ corresponds to the $j$th element of $d$ and 0 otherwise. Only the visible components of CTI descriptors can be matched. In that case, the matrix $Z{\cdot}A$ is used instead of $A$ and $d^{\mathrm{v}}$ instead of $d$ in computation of the optimal translation vector by (10).

If a SSC represents an occluded object in the scene, some of its vertices may not be the true object vertices. These vertices can produce false CTI descriptor components, which do not describe the true shape of the object. In order to assure that all descriptor components are computed from true object vertices, an additional visibility constraint is introduced. As previously explained, each $i$th component of a CTI descriptor, computed for a vertex set $V$, corre-

**Algorithm 3**
CHAL.

**Input:** ${}^{S}V_{S}$, ${}^{M}V_{M}$, $A$
**Output:** ${}^{S}T_{M,i}$, $i = 1, \ldots, n_{\mathrm{MSC}}$
1 : Compute the convex hull of ${}^{M}V_{M}$.
2 : Define a set of LRFs for this convex hull using the approach described in Section 5.1. Let us denote each of these LRFs by $L_i^{M}$, where $i = 1, 2, \ldots, n_{\mathrm{MSC}}$.
3 : Compute the convex hull of ${}^{S}V_{S}$.
4 : Define a LRF $L^{S}$ for this convex using the approach described in Section 5.1. The LRF is generated from the face cluster pair with the lowest cost.
5 : Compute the CTI descriptor
$$d_{S} = d(T({}^{S}V_{S}, {}^{L^{S}}T_{S})),$$
where
$${}^{L^{S}}T_{S} = \left[\begin{array}{c|c} {}^{L^{S}}R_{S} & 0 \\ \hline 0 & 1 \end{array}\right]$$
and $T({}^{S}V_{S}, {}^{L^{S}}T_{S})$ denotes the point set obtained by transforming ${}^{S}V_{S}$ with ${}^{L^{S}}T_{S}$.
6 : **For** $i \leftarrow 1$ **to** $n_{\mathrm{MSC}}$
7 : Compute the CTI descriptor
$$d_{M} = d(T({}^{M}V_{M}, {}^{L_i^{M}}T_{M})),$$
where
$${}^{L_i^{M}}T_{M} = \left[\begin{array}{c|c} {}^{L_i^{M}}R_{M} & 0 \\ \hline 0 & 1 \end{array}\right].$$
8 : $t \leftarrow (A^{T}A)^{-1}A^{T}(d_{M} - d_{S})$
9 : ${}^{S}T_{M,i} \leftarrow {}^{S}T_{L^{S}} \cdot {}^{L^{S}}T_{L_i^{M}} \cdot {}^{L_i^{M}}T_{M}$,
where
$${}^{L^{S}}T_{L_i^{M}} = \left[\begin{array}{c|c} I & t \\ \hline 0 & 1 \end{array}\right]$$
10 : **end for**
11 : **Return** ${}^{S}T_{M,i}$, $i = 1, \ldots, n_{\mathrm{MSC}}$.

sponds to a tangential plane in one of the vertices $p \in V$. The $i$th descriptor component is included in $d^{\mathrm{v}}$ only if the plane defined by $a_i$ and $d_i$ is tangential to the surface formed by the three planar patches meeting at the vertex $p$.

### 5.4. Hypothesis generation

By applying the CHAL algorithm to a particular SSC and a particular MSC, a set of object pose proposals is obtained. Each of these proposals represents a hypothesis that the considered SSC represents an instance of a target object in a particular pose with respect to the scene RF or part of this object. Such a hypothesis can be represented by a triple $h = (i, j, {}^{S}T_{M})$, where $i$ is the index of the SSC, $j$ represents the index of the target object model and ${}^{S}T_{M}$ represents the pose of the model instance on the scene. In the approach proposed in this paper, the CHAL algorithm is applied to all SSCs and to all models in the model database, resulting in a large number of hypotheses. These hypotheses are evaluated and pruned using the approach presented in Section 6.

## 6. Hypothesis evaluation and pruning

This section describes the process of selecting correct hypotheses from the hypothesis set generated by the methods presented in Section 5. Hypothesis selection is performed in three levels. In each level, the hypothesis cost is computed for every hypothesis and a certain number of hypotheses are rejected from further analysis according to this cost. After the third evaluation level, the final scene interpretation is generated, representing a consistent set of non-conflicting hypotheses.

### 6.1. Level 1: CTI matching and pruning of redundant hypotheses

On the first level of the hypothesis evaluation process, the hypotheses generated by the CHAL algorithm are evaluated by computing the cost for each hypothesis. High-cost hypotheses and redundant hypotheses are rejected. The CHAL algorithm generates hypotheses by aligning SSCs with MSCs, as explained in Section 5. The cost of a hypothesis generated by aligning a MSC with a SSC is based on the difference between their CTI descriptors. Let $d_{S}$ and $d_{M}$ be two CTI descriptors generated from a SSC and a MSC, respectively (lines 5 and 7 of Algorithm 3). The *CTI matching cost* is defined by

$$\delta_{\mathrm{CTI}}(d_{M}^{\mathrm{v}}, d_{S}^{\mathrm{v}}) = \frac{1}{n_{\mathrm{v}}} \sum_{i=1}^{n_{\mathrm{v}}} \min\left\{ \frac{\left(d_{M,i}^{\mathrm{v}} - d_{S,i}^{\mathrm{v}}\right)^{2}}{\sigma_{\mathrm{d}}^{2}}, 1 \right\}, \tag{12}$$

where $\sigma_{\mathrm{d}}$ is an experimentally determined constant. All hypotheses with the cost $\geq \tau_1$, where $\tau_1$ is an experimentally determined threshold, are excluded from further analysis.

In order to prevent generation of redundant hypotheses, the hypothesis set is pruned by non-maximum suppression described by Algorithm 2, where the CTI matching cost is used and the neighborhood of a hypothesis is defined as follows. A hypothesis $h' = (i', j, {}^{S}T'_{M})$ is a neighbor of a hypothesis $h = (i, j, {}^{S}T_{M})$ if they refer to the same model and the transformations ${}^{S}T_{M}$ and ${}^{S}T'_{M}$ have a similar translation and orientation. The translation is considered to be similar if

$$\left\| {}^{S}t'_{M} - {}^{S}t_{M} \right\|_{2} \leq \delta_{\mathrm{t}}. \tag{13}$$

The similarity criterion for the rotation is that the angles between the x- and z-axis of the rotation matrices ${}^{S}R_{M}$and ${}^{S}R'_{M}$ are $\leq \delta_{\mathrm{r}}$. Parameters $\delta_{\mathrm{t}}$ and $\delta_{\mathrm{r}}$ are experimentally determined constants. The merging step (line 8 of Algorithm 2) is omitted.

### 6.2. Level 2: hypothesis evaluation by image projection

Every hypothesis $h = (i, j, {}^{S}T_{M})$ which remained after the first level pruning is subject to the second evaluation level. At the second hypothesis evaluation level, the points of the model $\mathcal{M}_j$ associated with a hypothesis $h$ are transformed into the scene RF using ${}^{S}T_{M}$ in order to evaluate how well the scene points are explained by the transformed model points. Furthermore, in addition to explaining a subset of scene points, a correct hypothesis should be consistent with other scene points. We assume that the considered objects are not transparent and, therefore, model points should not occlude scene points which are visible in the depth image of the scene. Finally, assuming that all objects are placed on a planar surface, a hypothesis assuming that an object is positioned below the DPS is not consistent with the scene.

Since the evaluation described in this section is applied to a large number of hypotheses, computational efficiency of its implementation is critical. In order to reduce the number of operations required for the evaluation of each hypothesis, models are subsampled off-line using a voxel grid filter and this reduced point set is used for all computations described in this section. The critical parameter of this filter is the voxel size $s_f$, which defines the sampling resolution and hence the number of model points used for evaluation.

Two cues used for hypothesis evaluation in our approach are equivalent to cues i) and ii) applied in [7,8], with the exception of some implementation details described in this section. Let $\mathcal{M}_h^v$ be the set of model points transformed to the scene RF using ${}^S T_M$ which satisfy $n_p^T p < 0$, where $p$ is the vector of point coordinates in the camera RF and $n_p$ is the local surface normal of $p$. This condition requires that the model surface in the neighborhood of the considered point is oriented towards the camera, analogously to the visibility constraint explained in Section 5.3. Then, for every point $p \in \mathcal{M}_h^v$, the depth image point is determined on the same optical ray, i.e. with the same image coordinates. If this image point has a defined depth, then its 3D position $q_p$ and its normal $n_{q_p}$ are compared to $p$ and $n_p$. Hypothesis $h$ is assigned the *scene fitting score* computed by

$$\Omega(h) = \sum_{p \in \mathcal{M}_h^v} \omega(p), \tag{14}$$

$$\omega(p) = \begin{cases} \left(1 - \frac{\|p - q_p\|_2^2}{\rho_e^2}\right) \max\left(n_p^T n_{q_p}, 0\right), & q_p \text{ exists and } \|p - q_p\|_2 \le \rho_e \\ 0, & \text{otherwise} \end{cases}, \tag{15}$$

where $\rho_e$ is an experimentally determined threshold.

The scene fitting score can be falsely increased by matching the projected points of a false hypothesis to the points of the supporting planar surfaces. In order to avoid this, the points assigned to large planar surfaces by the approach described in Section 4.2 are excluded from the computation of the scene fitting score.

Another cue used for hypothesis evaluation is the *transparency cost* representing the number of transparent model points. If

$$z(q_p) \ge z(p) + \rho_t, \tag{16}$$

where $z(p)$ denotes the depth of the point $p$ and $\rho_t$ is a threshold, then the point $p$ is classified as a transparent point candidate. Because of the uncertainty of the model pose estimated by the CHAL, some model points close to the boundary of the model image projection can be falsely projected onto background points and therefore misclassified as transparent. In order to avoid this misclassification, the depth of $p$ should be compared not only to the depth of $q_p$ but also to the depth values in its neighborhood. However, examining a neighborhood for every point $p \in \mathcal{M}_h^v$ would be computationally expensive because of the large number of hypotheses evaluated on the second level. In order to achieve high computational efficiency, instead of examining an image neighborhood, we perform dilation of the depth image and examine only one point of the dilated depth image. The dilation is performed by assigning to each image point the minimum depth in its $w_d \times w_d$ neighborhood. The final decision whether a transparent point candidate $p$ is considered to be transparent or not is made using condition (16), where $z(p)$ is the depth value of the projection of $p$ onto the dilated depth image. The set of transparent model points is denoted in this paper by $\Phi_h$. The points of the DPS are not excluded from the computation of $\Phi_h$.

Analogously to the approach presented in [8], the object pose with respect to the DPS is used as the third cue for hypothesis evaluation. The cost which measures the consistency of a hypothesis $h$ with the DPS is determined by transforming all vertices of the convex hull of $\mathcal{M}_j$ using ${}^S T_M$ and computing the lowest vertex with respect to the DPS. The signed distance of this point from the DPS is taken as the measure of consistency of $h$ with the DPS. This distance is computed by

$$\delta_{DPS}(h) = \min_{p \in V_M} \left(q_{DPS}^T \cdot {}^S T_M \cdot \tilde{p}\right), \tag{17}$$

where $q_{DPS} = [\ n_{DPS}^T \quad -d_{DPS}\ ]^T$, $n_{DPS}$ and $d_{DPS}$ are the normal and the distance of the DPS with respect to the scene RF, respectively, $\tilde{p}$ is the homogeneous coordinate vector corresponding to a vertex $p$ and $V_M$ is the set of the model convex hull vertices. A negative value of $\delta_{DPS}(h)$ indicates that the lowest vertex of the predicted object is below the DPS. It is assumed that the objects cannot be positioned under the DPS. Hence, if $\delta_{DPS}(h) < -\tau_{PS}$, then $h$ is excluded from further analysis. The tolerance $\tau_{PS}$ is applied in order to compensate for measurement noise. If the object predicted by a hypothesis $h$ is placed on another object, then $\delta_{DPS}(h) > 0$. Such hypotheses are not rejected, but penalized, because it is assumed that this situation is less probable than the case where the object is placed on the DPS. The penalization term is defined by

$$f_{DPS}(h) = \min\left(\delta_{DPS}, \tau_{PS}\right). \tag{18}$$

The total *hypothesis evaluation cost* computed from the three described cues is

$$\Psi(h) = -\Omega(h)(1 - \gamma |f_{DPS}(h)|) + |\Phi_h|, \tag{19}$$

where $\gamma$ is an experimentally determined weighting factor.

A hypothesis list $\mathcal{H}_{ij}^{SM}$ is created for every pair SSC-model, where $i$ is the index of a SSC and $j$ is the index of a model. Every hypothesis $h = (i, j, {}^S T_M)$ is added to its corresponding list $\mathcal{H}_{ij}^{SM}$ and all lists are sorted according to the hypothesis evaluation score. Then, a hypothesis list $\mathcal{H}_i^S$ is created for every SSC and the top $m_{H21}$ hypotheses of every list $\mathcal{H}_{ij}^{SM}$, $j$ x003D= 1, ..., $n_M$ are added to $\mathcal{H}_i^S$. Notice that the list $\mathcal{H}_i^S$ contains at most $m_{H21}$ hypotheses associated with each model. Lists $\mathcal{H}_i^S$ are then sorted and the top $m_{H22}$ hypotheses of every list are considered for further analysis. However, if a large number of salient hypotheses are generated from a particular SSC, then these hypotheses are also preserved for further analysis. A simple measure of hypothesis saliency, which has shown to give good results, is the *normalized scene fitting score* computed by

$$\bar{\Omega}(h) = \Omega(h) / \left|\mathcal{M}_h^v\right|, \tag{20}$$

where $|\mathcal{M}_h^v|$ denotes the number of visible model points. A hypothesis $h$ is considered to be a salient hypothesis if $\bar{\Omega}(h) \ge \tau_2$, where $\tau_2$ is an experimentally determined threshold.

### 6.3. Level 3: fine-tuning by ICP and precise evaluation by image projection

On the third hypothesis evaluation level, all hypotheses resulting from the second level are fine-tuned using the ICP algorithm and a precise evaluation criterion is applied in order to distinguish between similar objects. For every hypothesis $h = (i, j, {}^S T_M)$, a point-to-plane ICP [39] is used to precisely align the $j$th model with the scene. High computational efficiency is achieved by using subsampled models, as explained in Section 6.2. The correspondences between the model and the scene points are determined by projecting the model to the depth image and identifying the

scene point $q_p$ on the optical ray of every model point $p \in \mathcal{M}_h^v$, as explained in Section 6.2. The number of ICP iterations is limited to 5, but the fitting procedure can stop earlier if the change in orientation between two subsequent iterations is $\leq 0.02$ rad. The points of the DPS are excluded from the fitting procedure. The result of the ICP fitting is a more precise model pose ${}^S T_M$.

After the ICP fitting, a new hypothesis evaluation cost is computed using an approach similar to the one described in Section 6.2. However, in order to be able to distinguish between very similar objects, a more precise variant of this approach is applied. The differences between the method used for computing the hypothesis evaluation cost at the second level and the one used at the third level are described as follows. Original, densely sampled models are used instead of subsampled models. A hypothesis $h = (i, j, {}^S T_M)$ is evaluated by projecting a transformed model $\mathcal{M}_j$ associated with a hypothesis $h$ to the depth image, where the z-buffering technique is used to determine a set of visible model points $\mathcal{M}_h^v$. This method is computationally more expensive than that used on the second level, but it is more precise because the z-buffering technique can handle self-occlusion, thereby producing a more correct set $\mathcal{M}_h^v$. Furthermore, instead of establishing correspondences between a model point $p$ and a scene point $q_p$ on the same optical ray, the closest point $q_p$ is searched for in a $3 \times 3$ neighborhood of the image projection of $p$. If no scene point closer than $\rho_e$ is found, then $p$ is tested for transparency using the approach applied on the second hypothesis evaluation level. However, instead of a dilated depth image, the original depth image is used, because the model pose adjusted by the ICP algorithm is expected to be more accurate than the pose estimated by the CHAL algorithm. Furthermore, small inaccuracies of the estimated pose are compensated by searching the $3 \times 3$ neighborhood.

The third hypothesis evaluation level penalizes hypotheses according to which a model instance covers only a portion of a convex surface detected in a scene. This is achieved by identifying the set $\mathcal{S}_h$ of all scene segments completely contained inside the convex hull of the model $\mathcal{M}_j$ transformed to the scene RF. Only the points belonging to the segments from the set $\mathcal{S}_h$ are considered in computation of the scene fitting score $\Psi(h)$. In order to compensate for measurement noise and uncertainty in the estimated model pose, the model convex hull is expanded by $\rho_e$.

After computing the new hypothesis evaluation cost for all hypotheses, all hypotheses with a positive cost $\Psi(h)$ and a low $\bar{\Omega}(h) < \tau_{FP}$ are also rejected in order to eliminate false positives. Finally, all lists $\mathcal{H}_i^S$ are sorted according to the new costs.

### 6.4. Using color information

In the case where the target objects are modeled by colored 3D meshes, the color of the model points is represented in the HSV color space and a 2D histogram of hue and saturation values is created for every model. The ranges of both the hue and the saturation value are divided into four intervals whereby a histogram of $4 \times 4$ bins is obtained. Because the hue value is poorly defined for low saturation colors, the points with saturation $< 4$ are excluded from the histogram, where the maximum saturation value is 255. In the testing phase, the same color histogram is computed for every SSC detected in the input depth image. The color histograms generated from a particular SSC are compared to the color histograms of all models associated with all hypotheses processed by the third hypothesis evaluation level. For each hypothesis $h = (i, j, {}^S T_M)$, the Bhattacharyya distance $\delta_{HS}(h)$ is computed between the color histogram of the $i$th SSC and the color histogram of the $j$th model. This distance is used as an additional cue in computing the hypothesis evaluation cost. In this case, the hypothesis evaluation cost is computed by

$$\Psi(h) = -\Omega(h)(1 - \gamma |f_{DPS}(h)| - \zeta \delta_{HS}(h)) + |\Phi_h|, \tag{21}$$

where $\zeta$ is an experimentally determined weighting factor. Furthermore, the distance $\delta_{HS}(h)$ is also used to prune hypotheses. The hypotheses with $\delta_{HS}(h) > \tau_{HS}$ are rejected.

### 6.5. Consistent scene interpretation

The final scene interpretation is generated by a greedy approach based on Algorithm 2, using the hypothesis evaluation cost computed on the third hypothesis evaluation level. Two hypotheses are considered to be neighbors if they are conflicting, i.e. if there is a subset of scene points which is explained by both hypotheses. The subset of image points explained by a hypothesis $h$ is the union of the segments contained in the set $\mathcal{S}_h$ identified on the third hypothesis evaluation level. Hence, two hypotheses $h$ and $h'$ explain the same subset of image points if $\mathcal{S}_h \cap S_{h'} \neq \emptyset$. However, since the applied depth image segmentation algorithm is not perfect, it is possible that some small, falsely detected convex surface is shared between two correct hypotheses. In that case, one of these two hypotheses would be falsely rejected. In order to avoid this, we consider a hypothesis to be consistent with a set of other hypotheses if a sufficient percentage of its supporting points is not explained by these hypotheses. This is implemented by introducing a set $S_Y$ of scene segments assigned to all hypotheses in the set $Y$ (See Algorithm 2). Instead of lines 6–11 of Algorithm 2, the following condition is evaluated

$$\frac{\sum_{C \in S_h \setminus S_Y} |C|}{\sum_{C \in S_h} |C|} \geq \tau_{free}, \tag{22}$$

where $|C|$ denotes the number of points in segment $C$ and $\tau_{free}$ is an experimentally determined threshold. If this condition is satisfied, then hypothesis $h$ is added to $Y$ and the set $S_Y$ is updated. Otherwise, hypothesis $h$ is rejected.

The result of the described procedure, which is also the final result of the proposed object recognition approach, is a set $Y$ of mutually consistent hypotheses representing the interpretation of the input depth image.

## 7. Experimental evaluation

In this section, a thorough experimental analysis of the proposed approach is presented. A contribution of the applied cues and sensitivity to some of the user-defined parameters is investigated. Object pose estimation accuracy is examined and the failure cases are discussed. The approach is compared to 9 state-of-the-art approaches.

### 7.1. Benchmark datasets and preprocessing

The proposed object recognition approach is evaluated on the following three datasets: the Kinect dataset [37], the Challenge dataset provided by Willow Garage and the dataset created at Imperial College London, referred to in this section as the ICL dataset [4]. The first two datasets contain 35 3D models of general-purpose objects present in the household, such as glasses, wine glasses, cups, detergent bottles, jugs, bowl, funnel, boxes, cans, etc. The Kinect dataset contains 50 RGB-D test scenes with a total of 176 object instances, while the Challenge dataset contains 176 test scenes with a total of 434 object instances. The models in the Kinect dataset are colorless. Since the main contribution of this paper is the novel approach for generating object proposals using depth information, this dataset is our primary target. This dataset is challenging because some of the models are very similar. Few sample models from the Kinect dataset are shown in Fig. 5[2]. In

[2] The models are preprocessed with Meshlab software in order to obtain uniformly sampled meshes.

**Fig. 5.** Sample models from the Kinect dataset.

some of the scenes, objects are occluded and cluttered. The Challenge dataset contains colored models. Some of the target objects of the Challenge dataset have identical or close to identical shapes and they can be distinguished only according to their color or texture. Although object recognition based on color is not the focus of our research, in order to expand our testing base and provide a comparison to a larger number of existing approaches, we augmented our object recognition system with a simple mechanism described in Section 6.4, which enables it to use color information. The ICL dataset contains six 3D models of target objects, i.e. a coffee cup, a shampoo bottle, a joystick, a camera, a juice carton and a milk bottle. A testing sequence of RGB-D images is provided for each of these target objects. The testing scenes of the Kinect and the Challenge dataset contain multiple randomly selected target objects. Hence, models of all 35 target objects must be stored in the model database of the object recognition system when the system is tested on these two datasets. This task is referred to in this section as the *multiple object detection* task. On the other hand, the ICL dataset is designed for testing *single object detection* approaches, i.e. every image of a particular testing sequence contains multiple instances of the same target object. Thus, in the case where the ICL dataset is used for testing, only one target object is contained in the model database of the evaluated shape instance detection algorithm for each of the six testing sequences.

The method used for detection of planar patches requires a uniformly sampled mesh as the input. Hence, the original 3D models contained in the Kinect dataset were preprocessed using the Meshlab software [40]. Furthermore, the models in the Challenge dataset are provided in the form of point clouds. Therefore, we had to generate meshes from these point clouds, which is also done using the same software. In the case of the ICL dataset, the target objects are modeled by dense triangular meshes suitable for our approach. Hence, no additional preprocessing was required.

The obtained triangular meshes were segmented into convex surfaces, as described in Section 4.1. For these convex surfaces, as well as for the whole models, LRFs were defined using the approach described in Section 5.1. For every LRF, a CTI descriptor was created. The total number of CTI descriptors created for the Kinect model database is 24 587, while for the Challenge dataset this number is 18 384. For the experiments with the ICL dataset, a single model database is created for each model, containing from 117 to 547 CTI descriptors.

Triangular meshes were also created from depth images of the scenes by the mesh construction algorithm presented in [32], which is included in the Point Cloud Library (PCL) [41]. The PCL was also used to compute normals of the input depth images.

### 7.2. Baseline configuration

The recognition performance of the proposed approach was tested by a series of experiments in which the pipeline presented in Sections 3, , and –6 was applied to the scene meshes. The values of the parameters used in these experiments are given in Table 1. These values are obtained through a series of preliminary experiments conducted during the development of the proposed method. Furthermore, after the algorithm had been completed, a series of experiments was performed in order to get an insight into the influence of different parameters on the algorithm performance. The parameter values presented in Table 1 represent the baseline configuration, which gives the best results. The same parameter values were used for all three datasets, except for the threshold $\tau_{\mathrm{HS}}$, which was adjusted for the ICL dataset, as explained in Section 7.7.

Object hypotheses are generated by matching the CTI descriptor of each SSC with the CTI descriptors of all MSCs. Hence, the total number of hypotheses generated using the CHAL algorithm is the product of the total number of SSCs and the total number of CTI descriptors generated from the model database. The maximum and average number of SSCs extracted from the test images of the Kinect and the Challenge dataset, as well as the maximum and average number of hypotheses generated by the CHAL algorithm per scene, are given in Table 2. Furthermore, Table 2 provides the number of hypotheses after each reduction step.

The total number of hypotheses generated by the CHAL algorithm is reduced to approximately 10% by rejecting all hypotheses with the CTI matching cost $< \tau_1$. The number of the remaining hypotheses is further reduced by rejecting approximately half of redundant hypotheses, as explained in Section 6.1. At the second hypothesis evaluation level described in Section 6.2, the number of hypotheses is further reduced by rejecting the hypotheses which predict objects whose lowest point is below a planar surface, i.e. hypotheses $h$ for which $\delta_{\mathrm{DPS}}(h) < -\tau_{\mathrm{DPS}}$. The number of hypotheses evaluated at the third hypothesis evaluation level is limited by

**Table 1**
Parameter values.

| parameter | value | parameter | value | parameter | value | parameter | value |
|---|---|---|---|---|---|---|---|
| $\tau_{\text{minC}}$ | 300 | $\tau_{\text{minPJ}}$ | 100 | $\rho_{\text{e}}$ | 0.03 m | $\tau_{\text{FP}}$ | 0.25 |
| $\sigma_{\text{noise}}$ | 0.005 m | $\tau_{\text{PS}}$ | 0.02 m | $\rho_{\text{t}}$ | 0.02 m | $\zeta$ | 0.5 |
| $\theta_{\text{FC}}$ | 10° | $\sigma_{\text{d}}$ | 0.01 m | $w_{\text{d}}$ | 17 | $\tau_{\text{HS}}$ | 0.25 |
| $\theta_{\text{LRF}}$ | 20° | $\tau_1$ | 0.6 | $\gamma$ | 10 | $\tau_{\text{free}}$ | 0.9 |
| $n_{\text{SC}}$ | 2 | $\delta_{\text{t}}$ | 0.01 m | $m_{H21}$ | 5 | | |
| $r_{\text{obj}}$ | 0.2 m | $\delta_{\text{r}}$ | 45° | $m_{H22}$ | 30 | | |
| $\tau_{\text{minPI}}$ | 10,000 | $s_{\text{f}}$ | 0.008 m | $\tau_2$ | 0.8 | | |

**Table 2**
Number of SSCs and hypotheses generated per scene.

| | Kinect | | Challenge | |
|---|---|---|---|---|
| | avg. | max. | avg. | max. |
| SSCs | 29 | 296 | 22 | 64 |
| hypotheses generated by CHAL algorithm | 717,940 | 7,277,752 | 409,941 | 1,176,512 |
| hypotheses with CTI matching cost $\geq \tau_1$ | 70,493 | 564,950 | 42,732 | 151,577 |
| non-redundant hypotheses | 35,655 | 266,814 | 22,188 | 64,256 |
| hypotheses with $\delta_{\text{DPS}}(h) \geq -\tau_{\text{DPS}}$ | 27,114 | 235,838 | 9715 | 52,492 |
| hypotheses evaluated at Level 3 | 653 | 7181 | 319 | 1367 |

parameters $m_{H22}$ and $\tau_2$. This number is given in the last row of Table 2. These hypotheses are evaluated using the most computationally expensive test described in Section 6.3, which includes the refinement of the estimated object pose by the ICP algorithm and hypothesis evaluation using the full model point clouds. This hypothesis set is pruned by the condition described in Section 6.3. The remaining hypotheses are used for the final scene interpretation generated by the greedy algorithm described in Section 6.5.

### 7.3. Comparison to other approaches and failure cases

Since the multiple object detection task is our primary subject, this section and Sections 7.4–7.6 report the results of the experimental evaluation using the Kinect and the Challenge dataset, while the single object detection task is considered in Section 7.7.

In order to evaluate the performance of the proposed object recognition approach in solving the multiple object detection task, we compared our approach to five state-of-the-art methods according to the precision, recall and F-score. The methods used for the comparison are presented in [8,37,42–44]. A hypothesis included in the final scene interpretation obtained by the proposed algorithm is considered to be a *true positive* if it predicts an object contained in the ground truth data assigned to this scene and the pose of this object is sufficiently similar to the ground truth pose. A similarity between an object pose estimated by the considered method and the corresponding ground truth pose is evaluated by transforming the object according to the associated pose and computing the *Root Mean Square Error* (RMSE) between the transformed model points and the corresponding scene points. The predicted object is considered to be a true positive if the RMSE is $\leq$ 0.03 m. All predicted objects included in the final result that are not true positives are declared as *false positives*. The number of ground truth objects not detected in the scene represent *false negatives*.

Table 3 presents the values of precision, recall and F-score for the proposed approach and all compared methods. The values reported for the compared methods are taken from [8]. From the results presented in Table 3, it can be concluded that the approach proposed in this paper has a significantly higher performance for the Kinect dataset, while the performance of our approach for the Challenge dataset is close to that of the state-of-the-art methods.

Two examples of correct scene interpretations are shown in the top row of Fig. 6. The proposed algorithm detected all target objects in the test scenes of the Kinect dataset and produced one false positive by interpreting a background object as a bottle. This false positive is the only error made by the proposed approach for the Kinect dataset. A similar error, shown in the bottom left image of Fig. 6, occurred in processing of the Challenge dataset, where a human hand is falsely interpreted as a target object. Although the criterion used to reject false positives, described in Section 6.3, resulted in a very high precision, these two failure cases indicate that there is still space for improvement. The other error which occurred in the processing of the Challenge dataset is a false detection of a can, shown in the bottom middle image of Fig. 6, which is mismatched with the other can of very similar shape, size and color. These two cans are shown in the bottom right image of Fig. 6. These two errors are the only errors made by the proposed algorithm for the Challenge dataset.

**Table 3**
Multi-object detection performance comparison.

| Dataset | Method | Precision | Recall | F-score |
|---|---|---|---|---|
| Kinect | Our | **0.994** | **1.000** | **0.997** |
| | GHV [8] | 0.970 | 0.915 | 0.942 |
| | Glover [42] | 0.894 | 0.864 | 0.879 |
| | Aldoma [37] | 0.909 | 0.795 | 0.848 |
| Challenge | Our | 0.995 | 0.998 | 0.997 |
| | GHV [8] | **1.000** | **1.000** | **1.000** |
| | Xie [44] | 1.000 | 0.998 | 0.999 |
| | Tang [43] | 0.987 | 0.902 | 0.943 |

### 7.4. Contribution of algorithm components and sensitivity to parameter values

In order to investigate the contribution of the planar surface assumption to the algorithm performance, we performed an experiment without using this assumption. The obtained results are shown in Table 4. From the comparison of these results to those obtained by the baseline configurations presented in Table 3, it can be concluded that the planar surface assumption improves the performance significantly. The most noticeable improvement is achieved in the case of the Challenge dataset, where the planar surface assumption increases the precision significantly by rejecting many false positives which predict objects below the DPS.

The contribution of the object pose refinement by the ICP fitting is investigated by an experiment in which the ICP fitting is not ap-

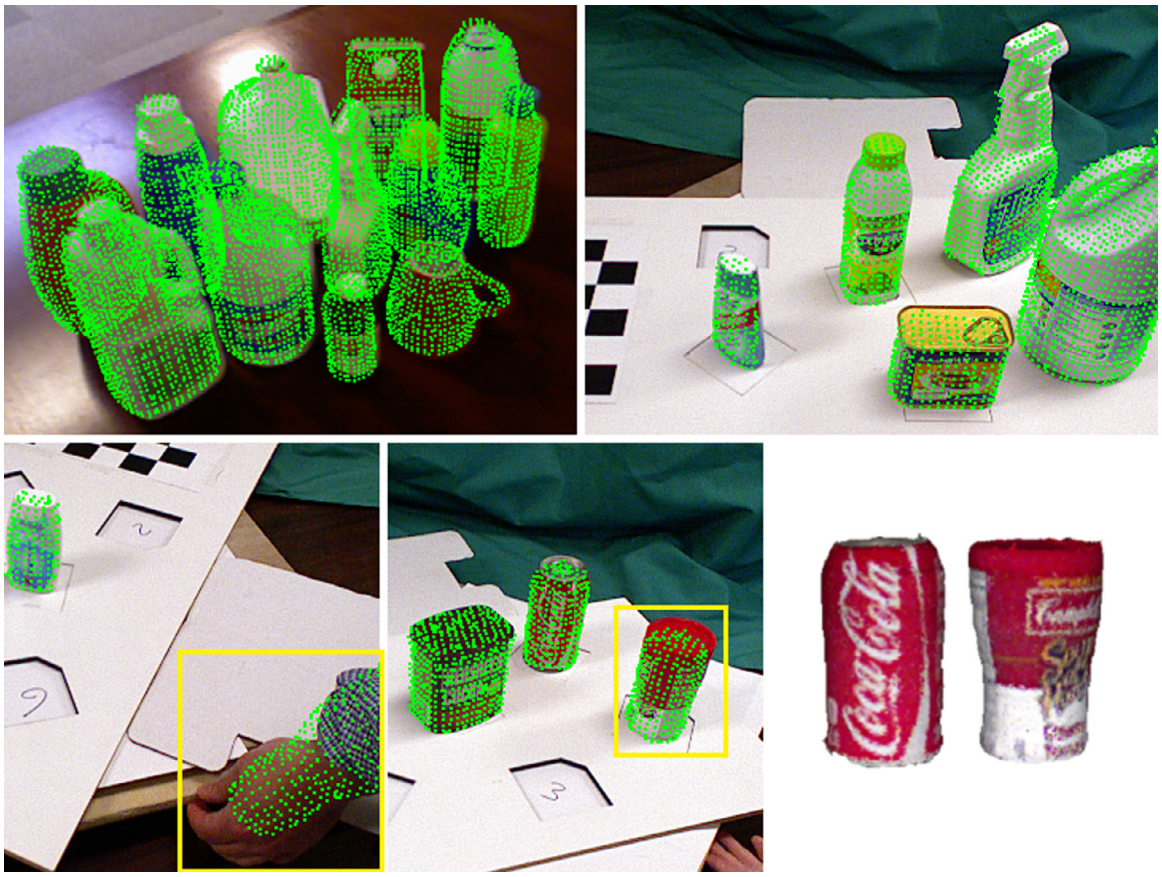

**Fig. 6.** Examples of successful object detection (top) and failure cases (bottom). The point clouds representing the detected objects are depicted by green dots. The failure cases are denoted in the bottom images by yellow rectangles. The mismatched can denoted by a yellow rectangle (bottom middle) is presented in the bottom right image together with the falsely detected can of a similar color, shape and size (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

**Table 4**
Contribution of the planar surface assumption, color information and pose refinement by the ICP algorithm.

| | Kinect | | Challenge | |
|---|---|---|---|---|
| | Precision | Recall | Precision | Recall |
| without the planar surface assumption | 0.927 | 0.932 | 0.625 | 0.940 |
| without using the ICP fitting | 0.902 | 0.886 | 0.954 | 0.956 |
| without using color information | – | – | 0.817 | 0.873 |

plied. Instead, the object poses estimated by the CHAL algorithm are used at the third hypothesis evaluation level. The obtained results are shown in Table 4. Although the object poses provided by the CHAL algorithm result in a relatively high precision and recall, the ICP fitting is necessary in order to achieve state-of-the-art performance.

The contribution of using color information is assessed by performing an experiment with the Challenge dataset without using color information. The obtained results are shown in Table 4. Since several objects of the Challenge dataset are very similar in shape, color information is crucial for distinguishing between these objects.

Furthermore, we performed a study of the influence of different parameter values on the algorithm performance by changing 7 parameters with respect to the baseline configuration in a wide range. Five experiments were performed for each parameter. The parameter values used in these experiments are given in Table 5. The results obtained with these parameter values for the Kinect and the Challenge dataset are presented in Tables 6 and 7, respectively. In the case of the Kinect dataset, the results obtained for all considered parameter values are better than the results obtained by the compared approaches, as reported in Table 3. In the case of the Challenge dataset, the color threshold $\tau_{HS}$ has the greatest influence, whose low values require a high similarity between the color histograms of the object in the scene and its corresponding

**Table 5**
Parameter values used in the sensitivity study.

| Parameter | Experiment | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| $s_f$ | 0.004 m | 0.006 m | **0.008 m** | 0.010 m | 0.012 m |
| $\gamma$ | 6 | 8 | **10** | 12 | 14 |
| $\tau_{FP}$ | 0.15 | 0.20 | **0.25** | 0.30 | 0.35 |
| $m_{H21}$ | 10 | 20 | **30** | 40 | 50 |
| $\tau_1$ | 0.4 | 0.5 | **0.6** | 0.7 | 0.8 |
| $\zeta$ | 0.3 | 0.4 | **0.5** | 0.6 | 0.7 |
| $\tau_{HS}$ | 0.15 | 0.20 | **0.25** | 0.30 | 0.35 |

**Table 6**
Precision/recall for the Kinect dataset obtained by different values of 5 parameters.

| Parameter | Experiment | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| $s_f$ | 0.977/0.977 | 0.989/0.994 | 0.994/1.000 | 0.983/0.989 | 0.994/1.000 |
| $\gamma$ | 0.989/0.994 | 0.989/0.994 | 0.994/1.000 | 0.989/0.994 | 0.977/0.983 |
| $\tau_{FP}$ | 0.983/0.994 | 0.989/1.000 | 0.994/1.000 | 0.994/0.994 | 1.000/0.983 |
| $m_{H21}$ | 0.983/0.983 | 0.989/0.994 | 0.994/1.000 | 0.989/0.994 | 0.989/0.994 |
| $\tau_1$ | 0.977/0.966 | 0.989/0.989 | 0.994/1.000 | 0.994/1.000 | 0.994/1.000 |

**Table 7**
Precision/recall for the Challenge dataset obtained by different values of 7 parameters.

| Parameter | Experiment | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| $s_f$ | 0.998/1.000 | 0.986/0.988 | 0.995/0.998 | 0.984/0.986 | 0.986/0.988 |
| $\gamma$ | 0.984/0.986 | 0.991/0.993 | 0.995/0.998 | 0.991/0.993 | 0.989/0.991 |
| $\tau_{FP}$ | 0.995/0.998 | 0.995/0.998 | 0.995/0.998 | 0.995/0.998 | 0.995/0.998 |
| $m_{H21}$ | 0.988/0.972 | 0.991/0.984 | 0.995/0.998 | 0.995/0.998 | 0.993/0.995 |
| $\tau_1$ | 0.998/0.995 | 0.995/0.998 | 0.995/0.998 | 0.995/0.998 | 0.995/0.998 |
| $\zeta$ | 0.993/0.995 | 0.993/0.995 | 0.995/0.998 | 0.993/0.995 | 0.989/0.991 |
| $\tau_{HS}$ | 0.986/0.947 | 0.993/0.979 | 0.995/0.998 | 0.989/0.995 | 0.986/0.995 |

**Table 8**
Execution time for all steps of the proposed approaches.

| Dataset | Time [ms] | $t_{pp}$ | $t_{pc}$ | $t_{L1}$ | $t_{L2}$ | $t_{L3}$ | $t_{SI}$ | Total |
|---|---|---|---|---|---|---|---|---|
| Kinect | average | 295 | 15 | 607 | 663 | 318 | 1.3 | 1900 |
| | maximum | 339 | 35 | 6343 | 6312 | 3924 | 44.3 | 16,928 |
| Challenge | average | 231 | 7 | 1214 | 232 | 229 | 0.2 | 1913 |
| | maximum | 269 | 13 | 2066 | 1270 | 1033 | 1.5 | 4312 |

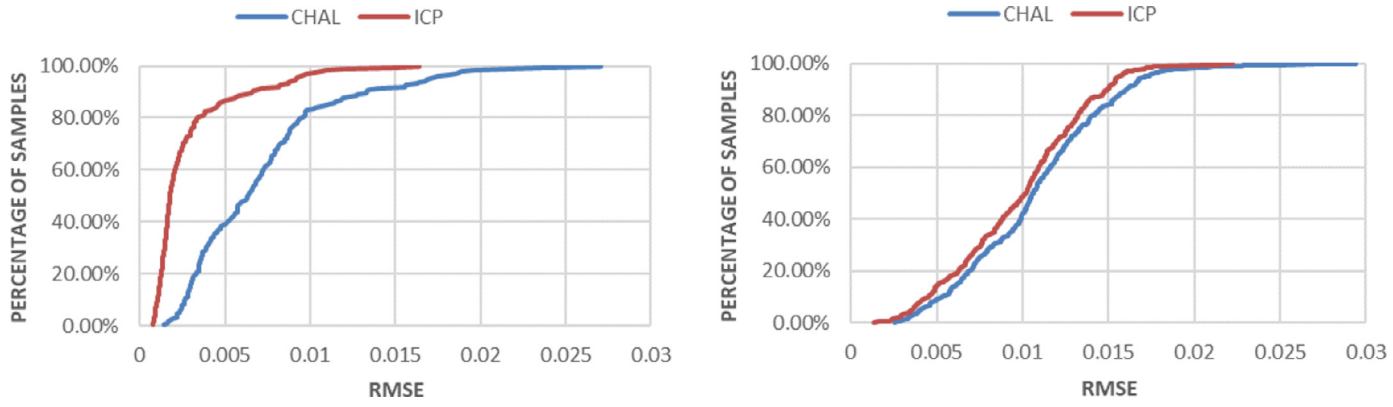


**Fig. 7.** The RMSE of the true positives generated by the CHAL algorithm (blue) and after pose refinement by ICP (red) for the Kinect dataset (left) and the Challenge dataset (right) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

model. Hence, if this value is too low, then many true positives will be falsely rejected.

### 7.5. Object pose estimation accuracy

The accuracy of object pose estimation achieved by the proposed algorithm is shown in Fig. 7. The RMSE values for all true positives are represented by a normalized cumulative histogram.[3] In the case of the Kinect dataset, 97% of true positives have the RMSE $< 0.01$ m after the ICP fitting. In the case of the Challenge dataset, the RMSE is higher, mainly due to less accurate ground truth data provided for that dataset. Furthermore, the object poses estimated by the CHAL algorithm give the RMSE $< 0.02$ m for 97% of true positives for both datasets. It can be concluded that the proposed CHAL algorithm produces highly accurate object pose proposals. Nevertheless, in order to distinguish between objects very similar in shape, the ICP algorithm must be applied at the final stage.

### 7.6. Execution time

The execution time of the proposed object recognition approach is given in Table 8. The experiments are executed on a PC with an Intel Core i7-4790 3.60 GHz processor and 16 GB of installed RAM memory running Windows 10 64-bit OS. In addition to the total execution time, the computation times required for different steps of the algorithm are also provided. The execution times of the following steps are reported: segmentation into planar patches ($t_{pp}$), aggregation of planar patches into planar and convex surfaces ($t_{pc}$), hypothesis generation and the first hypothesis evalu-

[3] A normalized cumulative histogram is a data representation where the horizontal axis corresponds to values of a measured variable x and the vertical axis represents the percentage of measurements which are $\leq$ x.

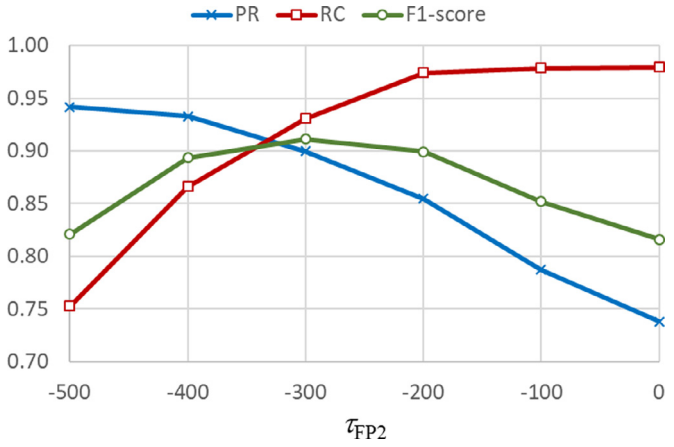


**Fig. 8.** Precision (PR), recall (RC) and F1-score curves of the proposed algorithm applied to the ICL dataset for different values of $\tau_{FP2}$.

ation level ($t_{L1}$), the second hypothesis evaluation level ($t_{L2}$), the third hypothesis evaluation level ($t_{L3}$), and generating the scene interpretation by greedy search ($t_{SI}$). In this table, maximum and average execution times for both datasets are presented. The time required for computation of normals is not considered in this analysis since this is a standard processing step for which many available tools exist.

From the execution times presented in Table 8, it can be seen that the average execution time of the proposed object recognition approach is below 2 s. The presented execution times are significantly shorter than the execution times of the three compared methods. The average execution time reported by Glover and Popovic [42] is 1 and 2 min, while the execution time reported by Aldoma et al. [37] is 10.4 s. The execution times reported by Tang et al. [43] and Xie et al. [44] are 20 s and 38.1 s respectively. For the GHV method [8], the authors have not reported the execution time.

The time saving achieved by the proposed three-level hypothesis evaluation strategy can be assessed by comparing the execution times of the three hypothesis evaluation levels with the number of hypotheses processed at each level, given in Table 2. For example, in the case of the Kinect dataset, the times required for the execution of the first and the second level are very similar, while the first level processes more than 20 times more hypotheses than the second level. Hence, it can be concluded that the first level saved a significant amount of processing time by rejecting many false hypotheses, while preserving true positives.

### 7.7. Single object detection with unmodeled objects in the scene

The testing scenes of the Kinect and the Challenge dataset, which we used to evaluate the performance of our algorithm in solving the multiple object detection task, contain mostly objects whose models are stored in the model database of the object recognition system, with accidental appearance of some background objects. This section presents the results of evaluation of our approach using the ICL dataset, whose testing scenes contain multiple instances of a single target object, but also multiple unmodeled objects intentionally inserted into the scene. In order to adapt our method to make it competitive with other methods designed to solve this task, we introduced an additional simple cri-

**Fig. 9.** Examples of successful object detection (top) and failure cases (bottom). The point clouds representing the detected objects are depicted by green dots. Two failure cases are denoted in the bottom images by yellow rectangles (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

**Table 9**
Single-object detection performance comparison.

| Object | LineMOD[3] | LC–HF[4] | Kehl [45] | Kehl [30] | Our (depth) | Our (depth+color) |
|---|---|---|---|---|---|---|
| Camera | 0.589 | 0.394 | 0.383 | 0.741 | 0.751 | **0.890** |
| Coffee | 0.942 | 0.891 | 0.972 | 0.983 | **0.985** | 0.963 |
| Joystick | 0.846 | 0.549 | 0.892 | **0.997** | 0.989 | 0.981 |
| Juice | 0.595 | 0.883 | 0.866 | 0.919 | 0.989 | **0.995** |
| Milk | 0.558 | 0.397 | 0.463 | 0.780 | **0.920** | 0.857 |
| Shampoo | **0.922** | 0.792 | 0.910 | 0.892 | 0.834 | 0.905 |
| Total | 0.740 | 0.651 | 0.747 | 0.885 | 0.911 | **0.932** |

terion for eliminating false positives. In Section 6.3., it is explained that the false positives are eliminated by rejecting all hypotheses with a positive cost $\Psi(h)$ and $\bar{\Omega}(h) < \tau_{\text{FP}}$. In the experiments with the ICL dataset, hypotheses with $\Psi(h) > \tau_{\text{FP2}}$ are rejected on the third hypothesis evaluation level. This additional constraint reduces the recall, but increases the precision of our algorithm resulting in the state-of-the-art performance on the ICL dataset. The obtained precision, recall and F1-score curves for different values of $\tau_{\text{FP2}}$ averaged over all six test sequences are shown in Fig. 8. The presented F1-scores are computed by performing an experiment with $\tau_{\text{FP2}} = 0$, rejecting all final hypotheses with $\Psi(h)$ greater than a particular threshold value and computing precision, recall and F1-score for that threshold value.

Two examples of successful object detection are shown in the top row of Fig. 9 and two typical failure cases are presented in the bottom row of this figure. In the case presented in the bottom left image of Fig. 9, the vertical surfaces of the denoted target object are oriented at a steep angle with respect to the camera, which results in holes in the depth image. These holes corrupt surface shape information resulting in a false negative. The shampoo bottle denoted by a yellow rectangle in the bottom right image of Fig. 9 is only partially visible. The proposed algorithm failed to detect this object because its visible part didn't provide sufficient information for reliable object detection. Nevertheless, in some cases the proposed approach is able to successfully recognize partially visible objects, like the joysticks in the top right image of Fig. 9.

We have compared our approach to four object recognition methods: LineMOD [3] and the approaches proposed by Tajani et al. [4] and Kehl et al. [30,45]. The results of this comparison are presented in Table 9. The values reported for the compared methods are taken from [30]. Even when relying on 3D geometry only, the proposed approach results in the best average F1-score in comparison to all other considered methods. The F1-score presented in this table is obtained for $\tau_{\text{FP2}} = -300$. Nevertheless, from the F1-score curve shown in Fig. 8 it can be seen that our approach achieves state-of-the-art performance for a wide range of $\tau_{\text{FP2}}$ values. The other parameters are set to the values presented in Table 1, which are also used for processing the Kinect and the Challenge dataset. It is interesting to compare these results obtained using only depth information to the approach proposed by Kehl et al. [30], which uses only RGB information for object detection.

Color information additionally improves the performance. The F1-score presented in Table 9 is obtained using $\tau_{\text{FP2}} = -100$ and $\tau_{\text{HS}} = 1$. In the case of the ICL dataset, a higher threshold $\tau_{\text{HS}}$ is used than in the case of the Challenge dataset. This threshold had to be relaxed in the case of the ICL dataset because the model colors do not accurately match the colors of the target objects in the testing scenes, probably because the models and the testing scenes are acquired in different lighting conditions. Furthermore, a higher threshold $\tau_{\text{FP2}}$ was used because the color term in the criterion (21) increases the cost $\Psi(h)$. Since the main contribution of this paper is related to object hypothesis generation and evaluation from the depth data, analysis of precision, recall and F1-score for different values of the thresholds $\tau_{\text{FP2}}$ and $\tau_{\text{HS}}$ applied when using color information is omitted.

## 8. Conclusion

The presented work addresses a very important and widely applicable problem of general-purpose shape instance detection. The research presented in this paper investigates the applicability of a novel technique for generating object proposals based on convex hull alignment in a standard object recognition pipeline. The presented experimental analysis demonstrates that competitive results can be achieved with a significantly different approach than those already proposed. Furthermore, we showed that high computational efficiency can be achieved by careful implementation and hierarchical application of standard hypothesis evaluation cues in a three-level hypothesis evaluation and pruning strategy. Each level reduces the number of hypotheses which must be processed by the next level, while keeping the correct hypotheses in the process. The proposed approach achieves state-of-the-art performance on a colorless dataset and superior computational efficiency.

The conducted experiments indicate that the assumption about objects on the scene positioned on a planar surface significantly contributes to the object recognition performance. Despite this assumption being valid for many applications, there are many possible applications for which it cannot be used. Instead of testing collision between predicted objects and a planar surface, a more general approach would be to penalize scene interpretations where the predicted objects are in collision with one another. A computationally efficient collision checking method, which could be used for that purpose, is our first choice for a future research topic.

Furthermore, the proposed approach generates object proposals from information about the 3D geometry of a given scene encoded in a depth image. It is capable of distinguishing between objects very similar in shape. However, in certain applications object color and texture could be more reliable cues. A combination of the approach proposed in this paper with different state-of-the-art color and texture descriptors is another future research topic which is worth investigating.

Finally, the current implementation of the proposed approach requires a certain number of user-defined parameters. A procedure which would determine some of these parameters automatically by optimization using an appropriate training dataset would significantly ease the adaptation of the presented approach to different applications and possibly improve the algorithm performance.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgment

This work has been fully supported by the Croatian Science Foundation under the project number IP-2014–09–3155.

## Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2020.107199.

## References

[1] S.H. Kasaei, A.M. Tomé, L. Seabra Lopes, M. Oliveira, GOOD: a global orthographic object descriptor for 3D object recognition and manipulation, Pattern Recognit. Lett. 83 (2016) 312–320, doi:10.1016/j.patrec.2016.07.006.
[2] A. Aldoma, F. Tombari, R.B. Rusu, M. Vincze, OUR-CVFH - Oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6DOF pose estimation, Lect. Notes Comput. Sci. (2012) 113–122 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 7476 LNCS, doi:10.1007/978-3-642-32717-9_12.
[3] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, V. Lepetit, Gradient response maps for real-time detection of textureless objects, IEEE Trans. Pattern Anal. Mach. Intell 34 (2012) 876–888, doi:10.1109/TPAMI.2011.206.
[4] A. Tejani, R. Kouskouridas, A. Doumanoglou, D. Tang, T.K. Kim, Latent-Class hough forests for 6 DOF object pose estimation, IEEE Trans. Pattern Anal. Mach. Intell 40 (2018) 119–132, doi:10.1109/TPAMI.2017.2665623.
[5] E.K. Nyarko, I. Vidović, K. Radočaj, R. Cupec, A nearest neighbor approach for fruit recognition in RGB-D images based on detection of convex surfaces, Expert Syst. Appl. 114 (2018) 454–466, doi:10.1016/j.eswa.2018.07.048.
[6] P. Durović, M. Filipović, R. Cupec, Alignment of similar shapes based on their convex hulls for 3D object classification, in: Proceedings of the IEEE International Conference on Robotics and Biomimetics, ROBIO, 2019, pp. 1586–1593, doi:10.1109/ROBIO.2018.8665154.
[7] C. Papazov, D. Burschka, An efficient ransac for 3D object recognition in noisy and occluded scenes, Lect. Notes Comput. Sci. (2011) 135–148 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 6492 LNCS, doi:10.1007/978-3-642-19315-6_11.
[8] A. Aldoma, F. Tombari, L. Di Stefano, M. Vincze, A global hypothesis verification framework for 3D object recognition in clutter, IEEE Trans. Pattern Anal. Mach. Intell 38 (2016) 1383–1396, doi:10.1109/TPAMI.2015.2491940.
[9] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, 3D object recognition in cluttered scenes with local surface features: a survey, IEEE Trans. Pattern Anal. Mach. Intell. 36 (2014) 2270–2287, doi:10.1109/TPAMI.2014.2316828.
[10] Z. Yu, Intrinsic shape signatures: a shape descriptor for 3D object recognition, in: Proceedings of the 12th IEEE International Conference on Computer Vision Workshops, ICCV Workshops, 2009, 2009, pp. 689–696, doi:10.1109/ICCVW.2009.5457637.
[11] H. Chen, B. Bhanu, Human ear recognition in 3D, IEEE Trans. Pattern Anal. Mach. Intell 29 (2007) 718–737, doi:10.1109/TPAMI.2007.1005.
[12] H. Liu, Y. Cong, C. Yang, Y. Tang, Efficient 3D object recognition via geometric information preservation, Pattern Recognit. 92 (2019) 135–145, doi:10.1016/j.patcog.2019.03.025.
[13] T. Fäulhammer, M. Zillich, J. Prankl, M. Vincze, A multi-modal RGB-D object recognizer, Proc. Int. Conf. Pattern Recognit. 0 (2016) 733–738, doi:10.1109/ICPR.2016.7899722.
[14] H. Ben-Yaacov, D. Malah, M. Barzohar, Recognition of 3D objects based on implicit polynomials, IEEE Trans. Pattern Anal. Mach. Intell 32 (2010) 954–960, doi:10.1109/TPAMI.2009.197.
[15] Z. Kuang, Z. Li, Y. Liu, C. Zou, Shape similarity assessment based on partial feature aggregation and ranking lists, Pattern Recognit. Lett. 83 (2016) 368–378, doi:10.1016/j.patrec.2016.05.026.
[16] S. Winkelbach, S. Molkenstruck, F.M. Wahl, Low-cost laser range scanner and fast surface registration approach, Lect. Notes Comput. Sci. (2006) 718–728 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 4174 LNCS.
[17] R.B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (FPFH) for 3D registration, (2009) 3212–3217. doi:10.1109/robot.2009.5152473.
[18] F. Tombari, S. Salti, L. Di Stefano, Unique signatures of histograms for local surface description, Lect. Notes Comput. Sci. (2010) 356–369 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 6313 LNCS, doi:10.1007/978-3-642-15558-1_26.
[19] Z. Yu, Intrinsic shape signatures: a shape descriptor for 3D object recognition, in: Proceedings of the IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, 2009, 2009, pp. 689–696, doi:10.1109/ICCVW.2009.5457637.
[20] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, J. Wan, Rotational projection statistics for 3D local surface description and object recognition, Int. J. Comput. Vis. 105 (2013) 63–86, doi:10.1007/s11263-013-0627-y.
[21] S.A.A. Shah, M. Bennamoun, F. Boussaid, Keypoints-based surface representation for 3D modeling and 3D object recognition, Pattern Recognit. 64 (2017) 29–38, doi:10.1016/j.patcog.2016.10.028.
[22] J. Yang, Q. Zhang, Y. Xiao, Z. Cao, TOLDI: an effective and robust approach for 3D local shape description, Pattern Recognit. 65 (2017) 175–187, doi:10.1016/j.patcog.2016.11.019.
[23] A.E. Johnson, M. Hebert, Using spin images for efficient object recognition in cluttered 3D scenes, IEEE Trans. Pattern Anal. Mach. Intell. 21 (1999) 433–449, doi:10.1109/34.765655.
[24] A.S. Mian, M. Bennamoun, R. Owens, Three-dimensional model-based object recognition and segmentation in cluttered scenes, IEEE Trans. Pattern Anal. Mach. Intell. 28 (2006) 1584–1601, doi:10.1109/TPAMI.2006.213.
[25] P.J. Besl, N.D. McKay, A method for registration of 3-D shapes, IEEE Trans. Pattern Anal. Mach. Intell. 14 (1992) 239–256.
[26] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, S. Savarese, DenseFusion: 6D object pose estimation by iterative dense fusion, (2019). http://arxiv.org/abs/1901.04780.
[27] S. Zhi, Y. Liu, X. Li, Y. Guo, Toward real-time 3D object recognition: a lightweight volumetric CNN framework using multitask learning, Comput. Graph. 71 (2018) 199–207, doi:10.1016/j.cag.2017.10.007.
[28] C. Ma, W. An, Y. Lei, Y. Guo, BV-CNNs: binary volumetric convolutional networks for 3D object recognition, in: Proceedings of the British Machine Vision Conference BMVC, 2017, 2017, doi:10.5244/c.31.148.
[29] X. Song, L. Herranz, S. Jiang, Depth CNNs for rgb-d scene recognition: learning from scratch better than transferring from RGB-CNNs, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, 2017, 2017, pp. 4271–4277.
[30] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, N. Navab, SSD-6D: making RGB-Based 3D detection and 6D pose estimation great again BT, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2017, pp. 1530–1538.
[31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, A.C. Berg, SSD: single shot multibox detector, Lect. Notes Comput. Sci. (2016) 21–37 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 9905 LNCS, doi:10.1007/978-3-319-46448-0_2.
[32] D. Holz, B. Sven, Fast range image segmentation and smoothing using approximate surface reconstruction and region growing, in: Proceedings of the International Conference on Intelligent System, 2012.
[33] R. Cupec, D. Filko, E.K. Nyarko, Segmentation of depth images into objects based on local and global convexity, in: Proceedings of the European Conference on Mobile Robots (ECMR), 2017, doi:10.1109/ECMR.2017.8098691.
[34] M. Attene, M. Mortara, M. Spagnuolo, B. Falcidieno, Hierarchical convex approximation of 3D shapes for fast region selection, in: Proceedings of the Eurographics Symposium on Geometry Processing, 27, 2008, pp. 1323–1332.
[35] K. Mamou, F. Ghorbel, A simple and efficient approach for 3D mesh approximate convex decomposition, in: Proceedings of the IEEE International Conference on Image Processing, 2009, pp. 3501–3504, doi:10.1109/ICIP.2009.5414068.
[36] R. Cupec, E.K. Nyarko, D. Filko, A. Kitanov, I. Petrovic, Place recognition based on matching of planar surfaces and line segments, Int. J. Rob. Res. 34 (2015) 674–704, doi:10.1177/0278364914548708.
[37] A. Aldoma, F. Tombari, L. Di Stefano, M. Vincze, A global hypotheses verification method for 3D object recognition, Lect. Notes Comput. Sci. (2012) 511–524 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 7574 LNCS, doi:10.1007/978-3-642-33712-3_37.
[38] J. Ritter, An efficient bounding sphere, in: A.S. Glassner (Ed.), Graphics Gems, Academic Press Professional, Inc., San Diego, USA, 1990, pp. 301–303.
[39] Y. Chen, Gérard Medioni, Object modeling by registration of multiple range images, in: Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, USA, 1991, pp. 2724–2729.
[40] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia, MeshLab: an open-source mesh processing tool, in: Proceedings of the 6th Eurographics Italian Chapter Conference, 2008, pp. 129–136, doi:10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.
[41] R.B. Rusu, S. Cousins, 3D is here: point cloud library (PCL), in: Proceedings of the International Conference on Robotics and Automation (ICRA), 2011, doi:10.1109/ICRA.2011.5980567.
[42] J. Glover, S. Popovic, Bingham procrustean alignment for object detection in clutter, in: Proceedings of the IEEE International Conference on Robotics System, 2013, pp. 2158–2165, doi:10.1109/IROS.2013.6696658.
[43] J. Tang, S. Miller, A. Singh, P. Abbeel, A textured object recognition pipeline for color and depth image data, in: Proceedings of the International Conference on Robotics and Automation, 2012, pp. 3467–3474, doi:10.1109/ICRA.2012.6224891.
[44] Z. Xie, A. Singh, J. Uang, K.S. Narayan, P. Abbeel, Multimodal blending for high-accuracy instance recognition, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, 2013, pp. 2214–2221, doi:10.1109/IROS.2013.6696666.
[45] W. Kehl, F. Milletari, F. Tombari, S. Ilic, N. Navab, Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation. Proceedings of the European Conference on Computer Vision (ECCV), 2016

**Robert Cupec** received his M.*Sc.* degree from the Faculty of Electrical Engineering, University of Zagreb, Croatia, in 1999, and Ph.D. degree from the Technische Universität München, Germany, in 2005. He is a full professor at the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek, J. J. Strossmayer University of Osijek, Croatia.

**Ivan Vidović** received his Ph.D. degree in 2016 from the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek, J. J. Strossmayer University of Osijek, Croatia. He is currently employed as a postdoctoral researcher at the same institution and his main research interest is computer vision with application to agriculture and robotics.

**Damir Filko** received his B.*Sc.* and Ph.D. degree in 2006 and 2013, respectively, both in electrical engineering from the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek. He is currently employed as an assistant professor at the Department of Automation and Robotics at same institution. His-current research interests include the application of computer vision in robotics and medicine.

**Petra Đurović** received her MEng degree in process computing from the Faculty of Electrical Engineering Osijek in 2015. In 2016, she enrolled in the postgraduate doctoral study program at the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek. Her research activity concerns robotics and robotic vision.