Computer Vision and Image Understanding 117 (2013) 1204–1214

Contents lists available at SciVerse ScienceDirect

Computer Vision and Image Understanding

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

Junction assisted 3D pose retrieval of untextured 3D models in monocular images

Hugo Álvarez *, Diego Borro

Applied Mechanics Department, CEIT and Tecnun (University of Navarra), Manuel de Lardizábal 15, 20018 San Sebastián, Spain

ARTICLE INFO

Article history:
Received 2 March 2012
Accepted 24 August 2012
Available online 26 November 2012

Keywords:
Computer vision
Object recognition
Shape
Geometric
Similarity measures

ABSTRACT

This article presents a comprehensive framework for the recognition of untextured 3D models in a single image. The method proposed here is capable of recovering a 3D pose in a few hundred of milliseconds, which is a difficult challenge using this type of model.

This proposal deals with 3D models that lack texture, so geometry features of the model are used as a basis of the 3D pose retrieval. An automatic process extracts the junctions and contours of the model, replacing the user interaction. Junctions will provide us an efficient mechanism to generate candidate matches, while contours will select the correct match based on a robust shape similarity evaluation. Our method only requires the 3D triangle mesh of the model as input, since the rest of the process is done automatically.

We demonstrate the behaviour of our approach against a variety of real scenes and models. Moreover, we explain how to face the first pose problem in a robust way using a history of votes. We also present a study of the method parameterisation, describing the influence of each parameter.

1. Introduction

The problem of 3D pose retrieval tries to recover the six parameters that define the viewpoint from which the model is shown in the image. The knowledge of these parameters would let us add virtual objects into the real world, giving the sensation that both worlds coexist.

3D pose retrieval in monocular images can be used by many computer vision applications, but it has achieved its highest popularity in industrial environments. It is used for multiple tasks, such as automatic inspection of model properties [23,30] or pick and place operations. Moreover, most of the models related to this environment share a common property: the lack of texture on their surface.

In this paper, we address the challenge of recognising an untextured 3D model from a monocular image, in the shortest amount of time and minimising the user interaction. This is a difficult task, since robust and fast techniques based on texture [1,14,15,21] cannot be used. Thus, the proposed approach is based on the geometry relationships between the junctions of a model. A junction is defined as a point where several edges meet, providing some geometric constraints. Therefore, the input model must be composed of a minimal set of junctions, such as the models presented in the experimental results.

The rest of the paper is as follows. In Section 2 some works related to 3D pose retrieval will be enumerated. Afterwards, the proposed 3D pose retrieval method will be detailed in Section 3. It is a complete framework that suppresses user interaction, building automatically all required geometry features. In Section 4 some experiments will be described, showing that the method achieves a good ratio between performance and robustness. Finally, in Section 5, the conclusions of the article will be cited and some future works will be proposed.

2. Related works

3D pose retrieval in monocular image is a well-known problem that has been studied by several authors, offering differents ways of proceeding according to the model and environment properties.

[1,14,15,21] use feature descriptors to determine a set of 2D-3D matches that define the 3D pose of the model. In the training step, some 2D views of the model are selected as representative. These views are processed, and features that lie on the surface of the model are extracted. Each feature is identified by its 3D position and a descriptor that codifies its neighbourhood. All of them are used to train a classifier that will be responsible of matching these features in the online phase. Moreover, descriptor-based techniques combined with geometry constrains [7] or linkage structures [22] improve precision and efficiency. All of them achieve good results and high frame rates, but the robustness is decreased for untextured objects. They are oriented to rich textured objects, since descriptors that can be extracted from their surface are more discriminative, thereby improving the matching results.

Other approaches use the contours of the model to get a positive match [8,9,16,27,29]. They are based on the shape of the

* Corresponding author. Fax: +34 943 21 30 76.
E-mail addresses: halvarez@ceit.es (H. Álvarez), dborro@ceit.es (D. Borro).

1077-3142/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.cviu.2012.08.012

model, and therefore, they are suitable for untextured models. In the training step, 2D views of the model are processed to extract the corresponding edges, transforming the online problem to a 2D–2D edge pattern matching. Although some of them only address the 2D–2D edge matching [16,27,29], they can be extrapolated to the 3D pose retrieval problem creating a similar training step. Furthermore, all of them need the acquisition of real 2D views. They are configured for specific light conditions and edge responses that are extracted from the real training images. This is not the case of [25,28], which solve this problem using a virtual training and building of 2D artificial views of the object, reaching more generality.

The third group of alternatives are based on geometric features [3,12,24]. In the offline-phase, distinctive 3D geometry features of the model are extracted. Furthermore, their geometric relationships are indexed in a data base. During the online-phase, image input features are detected and compared to the data base. Consistent matches will receive a vote, and the solutions with the highest number of votes are selected for further processing. Generally, these features provide poor distinctiveness, so an input feature generates multiple matches, increasing the computational effort. Due to the computational requirements, they are more popular for no time-critical applications, such as the 3D pose recovery in 3D scenes [4]. They have also been optimised using the correspondences specified by the user [6], but this has the disadvantage of requiring user intervention.

Our approach is related to geometric and contour based solutions. In the offline-phase, it builds a global hash table using the geometric relationships between the model junctions. These relationships are extracted from a set of 2D synthetic model views, which are created automatically, without user intervention. Moreover, a sophisticated geometric constraint between two junctions is proposed, which allows us to reduce false matches, and consequently, the computational time. This is the main difference with geometric alternatives, which spend too much time processing a high number of unnecessary matches. In the online phase, rather than using a voting scheme, matches with a similar position and scale are clustered. Each cluster is considered as a valid hypothesis and is evaluated using a similarity measure based on contours. This improves the robustness, as hypotheses are not thresholded based on the number of votes. Thus, following an idea similar to that of other authors [18,19], we combine multiple visual primitives to estimate the camera pose: optimised geometric constrains based on junctions are used to generate pose candidates, while contours are used to evaluate each pose candidate and select the correct one.

[28] is the work that comes closest to our 3D pose retrieval. Both studies create artificial 2D views of the model in the training step, use a robust contour based similarity measure, and both can recognise untextured 3D models in a reasonable amount of time. Compared to [28], our method offers less memory requirements and it is not limited to an offline predefined scale of the model. [28] stores the edges that are seen for each viewpoint and for each scale of the training phase, while our method only stores the edges and junctions that are seen for each viewpoint, without requiring space partition of the model scale (see Section 3.2.3). Considering that M_{ek} and M_{jk} are memory requirements to store the edges and junctions of the frame $k(M_{jk} \ll M_{ek})$ and that N_v and N_s represent the number of keyframes and the number of scales, we can state that $\sum_{k=1}^{N_v}(M_{ek}+M_{jk}) < \sum_{k=1}^{N_v * N_s} M_{ek}$.

The use of junctions for object recognition is not a new idea. They have demonstrated their viability for 2D object recognition [32]. However, an expensive edge detection algorithm and complex descriptors are required to get successful results, which takes several minutes. Thus, compared with this solution, our technique offers an efficient and robust mechanism to handle junctions and recognise 3D objects in a few hundred milliseconds, which is fast enough for time-critical applications.

3. 3D pose retrieval

The proposed algorithm overview is shown in Fig. 1. It is divided into two main blocks. In the first one (preprocessing stage) geometry features of the model are extracted and used to build a collection of virtual keyframes. Moreover, these keyframes are used by the second block pipeline (execution stage), which process the camera image to find positive correspondences between camera image features and virtual keyframe features. These positive correspondences will provide us the necessary data to establish the object presence in the current view. We give an in depth explanation of each block in the following sections.

3.1. Preprocessing stage

This stage involves the extraction of all the information that the model provides us, overriding completely the user interaction. Nevertheless, this aim is made difficult by the fact that the model lacks texture, which would let us apply a well known method from the literature [1,14,15,21]. Thus, the only available information that the model retains is its geometry configuration, which will be enough to achieve automatic object detection. More precisely, the proposed method only requires the 3D triangle mesh of the object as an input (provided as an *obj-file* for example). This stage is executed only once per model.

3.1.1. Geometry feature extraction

The detection of geometry features (such as edges) from a triangular mesh is not a new problem [10,20,33]. Most techniques are based on the detection of sharp changes between the normals of neighbour triangles, with some local constraints to improve the results [10,33]. Other authors, however, render the model from different points of view and back project detected 2D edges to 3D space, taking into account the texture of the model [20].

In our work, a simplified version of [10,33] has been implemented. Thus, a 3D sharp edge is detected if the angle between its neighbour facets is larger than a predefined threshold (40° for all our experiments), without considering local constraints. Furthermore, we also detect junctions in the 3D model. If two 3D sharp edges have one coincident vertex and the angle that they form is between a predefined minimum and maximum thresholds (80° and 100° respectively for our experiments), then both of them

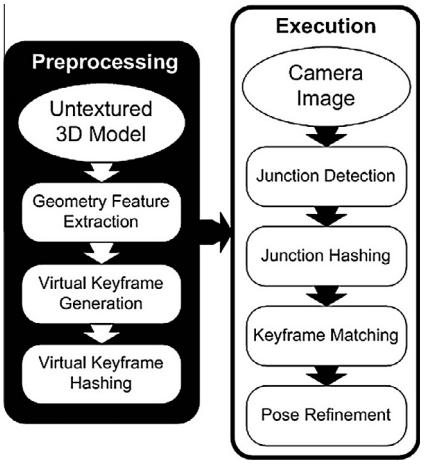

Fig. 1. Algorithm overview.