Automatic 3D Car Model Alignment for Mixed Image-Based Rendering

Rodrigo Ortiz-Cayon[1,*], Abdelaziz Djelouah[1,*], Francisco Massa[2], Mathieu Aubry[2], George Drettakis[1]

[1]INRIA [2] Ecole des Ponts ParisTech

Abstract

Image-Based Rendering (IBR) allows good-quality free-viewpoint navigation in urban scenes, but suffers from artifacts on poorly reconstructed objects, e.g., reflective surfaces such as cars. To alleviate this problem, we propose a method that automatically identifies stock 3D models, aligns them in the 3D scene and performs morphing to better capture image contours. We do this by first adapting learning-based methods to detect and identify an object class and pose in images. We then propose a method which exploits all available information, namely partial and inaccurate 3D reconstruction, multi-view calibration, image contours and the 3D model to achieve accurate object alignment suitable for subsequent morphing. These steps provide models which are well-aligned in 3D and to contours in all the images of the multi-view dataset, allowing us to use the resulting model in our mixed IBR algorithm. Our results show significant improvement in image quality for free-viewpoint IBR, especially when moving far from the captured viewpoints.

1. Introduction

Image-Based Rendering (IBR) is emerging as a viable approach for free-viewpoint navigation in captured environments, thanks to advances in camera calibration [48] and multi-view stereo [18, 25], as well as the rendering algorithms themselves [7, 34]. A key element of a high quality IBR is good 3D reconstruction estimated from the images. While great progress has been made in this domain, these methods do not work well in the case of transparent surfaces or reflective objects. IBR methods try to compensate for missing 3D geometry using strategies such as front-parallel assumptions [57] or 2D image warps [7]. For poorly reconstructed foreground objects, closer to the camera, this is usually not sufficient to mask errors in the reconstruction. A typical example for outdoors scenes are cars (Fig. 1).

We focus on urban environments, where captured scenes contain buildings and many man-made objects (cars, signposts, benches etc.). With the recent development of 3D model databases it is more and more likely to find a corresponding model to objects in a captured scene. Trying to use these CAD models is a legitimate strategy that has been used for various application such as depth correction [29] or image editing [26], even though selected 3D models usually do not exactly correspond to the images, requiring deformation of the model. However, these previous methods rely entirely on user interaction for selection, placement, alignment and deformation of the model and often are not designed to handle multi-view data which is required for IBR.

We present an *automatic* method which leverages databases of 3D CAD models for better IBR quality. The core idea is to use the stock models as a better proxy for the objects in the scene. To be visually convincing the stock models first need to be correctly placed in the scene and carefully aligned with the silhouettes in the images. We then use the model in our mixed IBR algorithm by a two-pass algorithm blending the background with the selected object. Our approach builds on learning methods in a preprocess and proposes an improved contour-based alignment and morphing approach to *automatically* chose, align and morph 3D models in a reconstructed scene for IBR. To the best of our knowledge this is the first automatic method for this process to improve IBR.

Our contributions can be summarized as follows. We first adapt learning-based methods to detect and identify an object class and pose in images. We then propose a method which exploits all available information, namely partial and inaccurate 3D reconstruction, multi-view calibration, image contours and the 3D model to achieve accurate object alignment for morphing. Our method provides fine-grain alignment and automatic correspondence detection for contours, achieving a good initial placement in the scene. Thanks to the good initial placement and the correspondence detection, we can automatically morph the stockmodel to better align with contours in all the images of the multi-view dataset. The resulting model is then used in our mixed IBR algorithm, greatly improving image quality, especially

*Authors contributed equally.

978-1-5090-5407-7/16 $31.00 © 2016 IEEE
DOI 10.1109/3DV.2016.37

when moving far from the captured viewpoints.

Our approach is fully automatic and can directly benefit from any future improvement in model selection or larger databases. We demonstrate our method on the example of cars in urban environments, since a sufficiently large 3D model database for cars is available [6]; when databases of models for other objects become available, our approch can be directly applied. Our experiments demonstrate a clear improvement in rendering quality compared to state of the art methods.

2. Previous Work

We propose a new mixed IBR algorithm, improving rendering quality of objects by automatically retrieving, aligning and morphing 3D geometry from stock models. Since each of these components is a vast area in itself, we restrict discussion of previous work only to the most closely related methods.

2.1. Image-Based Rendering

Image-based rendering is an effective method to synthesize highly realistic virtual imagery of captured scenes, since it includes all real-world geometric and appearance detail, complex materials and illumination effects. Initial methods [31, 20] required a very dense and structured set of images and provided only very limited displacements from the captured viewpoints. Other methods either manually create an approximate 3D representation of the scene [13] or use sophisticated blending [4]. The last decade has seen the development of automatic camera calibration [48] and improvements in the quality of 3D reconstruction from multi-view image datasets [19, 25], making IBR a reasonable option for visualization of the urban scenes we target, despite only moderate accuracy and completeness of the 3D reconstruction.

More recently, several IBR methods [7, 34, 38] use image warping or image correspondences with high quality results for facades or the "backdrop" of urban scenes. However, foreground objects are often reflective (e.g., cars), and are thus hard to reconstruct. Such methods still result in severe visual artifacts for these objects when moving away from the input cameras.

In contrast, we provide an end-to-end automatic pipeline which finds a good match for such objects from stock 3D models, and then performs fine-grain alignment and morphing to provide *multi-view* consistent 3D model for the object, resulting in much higher visual quality for free-viewpoint IBR.

2.2. 3D Object Databases and Learning

Initial methods to align 3D models to photos extract edges from both the 3D model and the photograph [43, 23, 35]. Such approaches work well especially for untextured objects [32, 1], but can be limited by the difficulty of extracting reliable and consistent edges in 2D and 3D. The most popular approach for matching CAD models to photograph is to use handcrafted descriptors such as HOGs [10, 50, 2] or learned Convolutionnal Neural Network (CNN) features [36, 3]. Considering the recent success of CNN features, we follow this last option.

CNNs have demonstrated impressive results in many domains such as image classification [28, 47]. In addition, they provide intermediate features which have proven to be generic enough to be re-used or adapted for very different tasks [54, 17]. Here we use CNNs to first detect object categories using the framework of [16] and then simply match images and rendered views from the 3D models using the intermediary *pool4* features of the network of [47] to retreive the corresponding 3D model in a manner similar to [3]. As mentioned before, here we focus only on cars to validate our approach since this is the most complete database available for objects in the scenes we target.

We next review previous methods for car pose estimation. Koller proposed a method [27] based on a polyhedral 3D vehicle model which was further improved by Ferryman *et al* [14]. The use of more complex deformable models has become more common [30]. Hodlmoser *et al* [22] applied Random Forests (using handcrafted features) to track cars in videos. More recently, fine-grained model classification has received more attention [33]. By taking advantage of 2D detections from deformable part models, a 3D model is deformed to better fit the estimated landmark positions. Part based features are then used for fine grained model classification. Zhu *et al.* [56] learn a compact representation of objects based on the detection of discriminative parts. 3D shape and pose are then inferred from a single image. Finally, Mottaghiusing *et al* [37] use a coarse-to-fine hierarchical representation for object detection, pose estimation and sub-category recognition. This method is not limited to cars and results are demonstrated with planes and boats.

Model selection and pose estimation are the first component of our solution; To be as generic as possible we build on the CNN-based methods and extend them to handle the multi-view data we treat here.

2.3. Geometry Alignment

A first category of methods treat automatic 3D model alignment. Earlier solutions [44, 12, 41] were based on a level set formulation and assumed a fixed 3D model. To handle different types of 3D models, a common approach is to learn an embedding into a lower dimensional space using kernel principal component analysis [11] or Gaussian process latent variable models [40]. Recent methods take

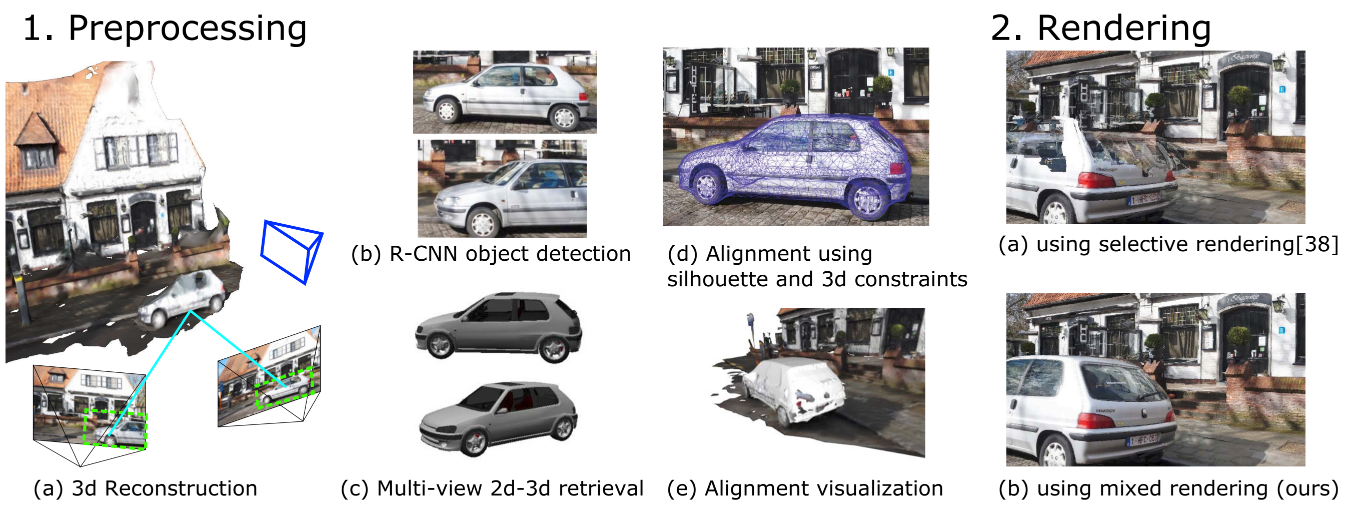
1. Preprocessing
2. Rendering
(a) 3d Reconstruction
(b) R-CNN object detection
(c) Multi-view 2d-3d retrieval
(d) Alignment using silhouette and 3d constraints
(e) Alignment visualization
(a) using selective rendering[38]
(b) using mixed rendering (ours)

Figure 1. **Overview.** We propose a new method for Image based rendering that takes advantage of 3D model databases. At a fully automatic *Preprocessing* stage, (1.a) cameras are calibrated and 3D reconstruction is estimated using multi-view stereo [25]. Objects of interest (here cars) are detected using R-CNN[16]. These detections are matched using viewing rays in cyan. (1.b) The corresponding images are used to query our database of 3D objects, (1.c) which returns a 3D model and an orientation relative to the cameras. (1.d) Combining depth and silhouette cues in multi-view, (1.e) a better alignment of the object is obtained. During *Rendering*, we generate the novel viewpoints illustrated by the blue camera (1.a). State of the art methods (e.g., [38]) exhibit strong artifacts on the car (2.a). Using our approach (2.b) we achieve good rendering quality for both the car and the background.

advantages of these dimensionality reduction approaches to estimate 3D shape, 2D-3D pose and image segmentation [46, 42]. Relying on dimensionality reduction, the unique parts present in a few models may disappear from the generic model. Our solution takes advantage of the diversity of models in the dataset and finds the closest one. It can also directly benefit from any improvement in matching or any new instance added to the database.

In CG applications, 3D models are fitted to images for various applications such as image edition [39] or 3D modeling [53]. In all cases user input is required for 3D geometry alignment. Zheng *et al* [55] rely on user interaction to help approximate underlying geometry with cuboids. A similar approach is adopted in [8] but the user has access to richer 3D components. In [26], the user selects a very similar 3D model which is deformed to fit the image according to user provided constraints. The main advantage of this method is its ability to generate novel viewpoints of the object. The closest method related to our work is the depth estimation method using 3D models [29] by Lee *et al*. Here the objective is to help the 2D-3D conversion for images and videos. Using a model aligned with the input image, depth is coherently inpainted. However the process heavily relies on manual user interaction at all stages: model selection, initial constraints for pose estimation and image segmentation for morphing. In contrast, our approach provides a fully automatic pipeline for all these steps, leading to a solution that is scalable and thus usable for IBR.

3. Overview

The goal of our approach is a high-quality mixed Image-Based Rendering algorithm for urban scenes, taking advantage of recent advances in object detection/recognition and the ever growing 3D object databases.

Input. The input to our approach is a set of photographs of the scene. We obtain calibrated cameras and an approximate geometry of the scene (proxy), using structure from motion (VisualSfM[51][52]) and multi-view stereo reconstruction (CMPMVS[25]); other alternatives could be used for this step. We also use the ShapeNet 3D object database [6] for the object retrieval step.

Object Selection and Preprocessing. Object bounding boxes are obtained using a recent detection algorithm [16]. We use the 3D geometry to put the detections into correspondence (Fig. 1.a). These images are used to find the corresponding 3D model and its orientation with respect to the cameras (Fig. 1.c).

Object Alignment. After obtaining the 3D models, we place the object in the scene. We use a multi-view approach taking advantage of the detection bounding boxes (5.1), the available geometry and silhouette matching (5.2).

Object Morphing and Rendering. After the alignment, the object mesh can still be different from the actual object geometry. In this case, we use morphing based on silhouettes to obtain a closer fit. Rendering is achieved by compositing background rendering using superpixels [38] and a ULR-like [4] rendering. Results clearly demonstrate the advantage of this approach (Fig. 1.2.b).

4. Stock 3D Model from Multi-view Images

The first step of our algorithm is to identify the regions in the input images containing the objects of interest, then select the stock 3D model and finally create a model which is suitable for further processing.

4.1. Multi-View Object Class Detection

We use the multi-region R-CNN [16] – one of the top-performing detection algorithms. The original method treats each input image of the dataset independently and produces candidate bounding boxes corresponding to the objects requested (e.g., car, chair etc.).

For each input view, we run the R-CNN, which provides a set of candidate 2D bounding boxes (Fig. 2). To put these bounding boxes into correspondence we rely both on appearance and geometry. We first cluster candidate regions based on appearance. Candidates belonging to different color clusters should never be matched. For each pair of object candidates – from different views – we compute the intersection of the viewing line passing through the center of the 2D bounding boxes (Fig. 1.a). The largest 3D-point clusters identify the objects in the scene and match 2D detections. After this step we have a set of candidate objects with corresponding images from different viewpoints. Next we use this data to find the corresponding stock models.

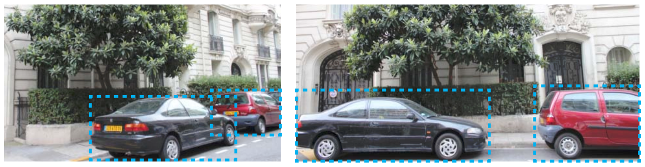
Figure 2. **Object detection.** Using the region aware detection algorithm [16] we obtain tight 2D bounding boxes of the objects of interest in the scene.

4.2. Multi-View 2D-3D Retrieval

We use the ShapeNet [6] database to find stock models using the 2D bounding boxes from all views. Currently, ShapeNet has a rich collection of the class “car” which we use to validate our approach. We downloaded 5K car models from this database and for each 3D model we rendered the object from 108 viewpoints of the viewing sphere, with azimuth and elevation increasing 10 degrees in the range of $[0, 360)$ and $[0, 30)$ respectively (see Fig 3). This constitutes our database of 5K car models, each associated to 108 views of the object.

We next use the images obtained from the bounding boxes to query our object database. Following Massa et al. [36], we compare the images using the cosine distance on *pool4* features. We tested features of AlexNet [28], VGG [47] as well as their adapted version using the framework of [36], which aims at bridging the domain gap between rendered 3D models and photographs. As expected, we found that the retrieval using AlexNet features were of lower quality than using the more powerful VGG features. More surprisingly, we also fond that the retrievals with adapted features were of lower quality, probably because the adaptation layer of [36] also destroys some image information.

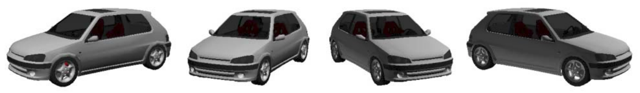
Figure 3. **Example of Renderings**. Four out of 108 views generated for one car model of the ShapeNet database.

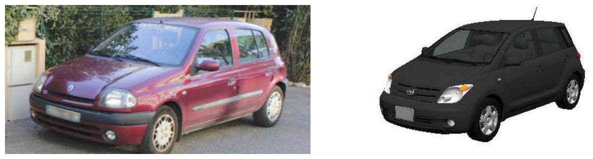
Figure 4. **Model and orientation matching.** The detection algorithm provides 2D bounding boxes for the object. After cropping, the images are used to query the database. Right: the matched rendering which provides both a 3D model and an orientation.

Each query is matched with an image from the database. This gives both a 3D model ID and an orientation with a matching score (of the object in the scene). Orientation is expressed as azimuth-elevation angles (θ, ϕ) with respect to a camera viewing the object at the origin. Because we have access to several views of the same car, we were able to further refine the retrieval using this information. Interpreting the comparison score between CNN features as a log probability, we compute a single score for each 3D model by simply summing the maximum score for this model for each of the unoccluded views of the model. Typical results of this step are shown in Fig. 4.

5. Positioning the Mesh

We now have a mesh corresponding to each object identified in the images of our multi-view dataset. The rendering with the highest matching score gives the orientation of the model with respect to one camera. To place the mesh in the scene we follow a multi-view strategy taking advantage of the detection information from all the views. To achieve more precise alignment we use both reconstructed geometry and silhouette matching. The final output is a set of rigid transformation parameters Λ corresponding to scale, translation and rotation for each 3D model.

5.1. Initial Pose Estimation

To align the 3D model obtained from the previous step (Sec. 4.2), we rely on the matched image with the highest score from the database. We know the orientation of the

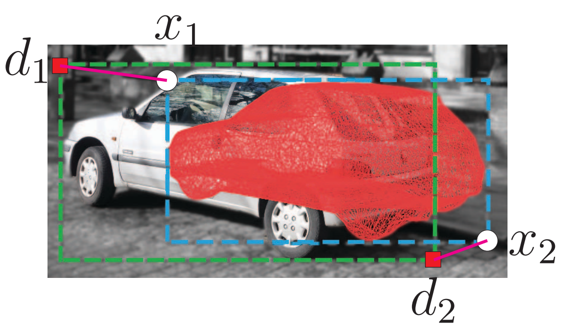

Figure 5. **Constraints from detection 2D boxes**. To have a better starting point for object alignment, we use the constraints from the detection bounding boxes. The bounding box corners x_1 and x_2 of the mesh should match the corners d_1 and d_2 of the R-CNN detection box.

3D models with respect to the corresponding virtual camera C_r, which we represent as rotation R_r. This rotation is not sufficient to align the 3D model, since our database consists of images rendered with the object at the center (Fig. 3) while in real world images the object can be present at different locations (Fig. 2). To compensate for this, we compute the rotation matrix that aligns the camera central axis ray with the viewing line passing by the center of the detected bounding box. After the rotation, a first estimate of the car position is computed as the point that minimizes the sum of squared distances to all rays casted through 2D bounding boxes.

We further improve this first estimate of the transform parameters Λ by leveraging the 2D bounding boxes from object detection (Fig. 5). By minimizing the distance between detection bounding boxes and the bounding boxes from 3D projection, we obtain a better initial estimate of the rigid transform parameters Λ:

$$\Lambda^* = \arg\min_{\Lambda} \sum_{i=1}^{n} dist(x_1^i - d_1^i)^2 + dist(x_2^i - d_2^i)^2 \quad (1)$$

where $\{x_1, x_2\}$ and $\{d_1^i, d_2^i\}$ respectively define, for view i, the sets of upper left and bottom right corners of the 2D bounding boxes of the object and the detection. The distance function $dist$ is defined as

$$dist(x, x^d) = \begin{cases} 0 & \text{if } x \text{ or } x^d \text{ on image border} \\ \|x - x^d\|_2 & \text{otherwise} \end{cases} \quad (2)$$

We use gradient descent to estimate this initial set of parameters Λ. We provide the full formulas for the partial derivatives in supplemental material.

5.2. Multi-View Alignment

After the initial pose estimation, the model is placed in the correct general region of the 3D scene. It now becomes possible to further improve the alignment of the 3D model

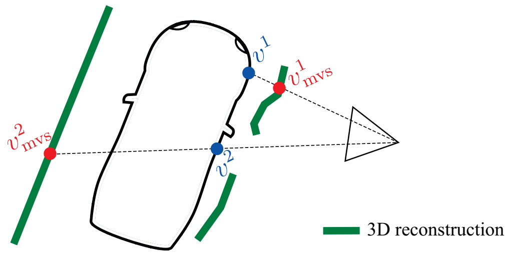

Figure 6. **Constraints from 3D reconstruction.** For each camera, the line of view for object vertices v^1 and v^2 intersects the available 3D reconstruction (in green; note that part of the car is missing) at 3D points v^1_{mvs} and v^2_{mvs}. In this case, the constraint from v^2_{mvs} is ignored.

using contours and available 3D reconstruction. We continue to use Λ to indicate pose parameters (rotation, translation and scale). We model the alignment step by solving the following optimization problem:

$$\Lambda^* = \arg\min_{\Lambda} E(\Lambda) = E_{3D}(\Lambda) + E_{edge}(\Lambda) \quad (3)$$

The first term (E_{3D}) corresponds to distance between the partial 3D reconstruction from MVS and the mesh. The second term (E_{edge}) tries to align the silhouette of the mesh with the corresponding edges in the images.

Constraints from partial 3D reconstruction. Multi-view stereo algorithms [25, 15] provide a coarse 3D reconstruction of the object. This reconstruction often contains inaccuracies and holes, typically in regions containing windows or strongly reflective surfaces where depth estimation from stereo matching algorithms is unreliable. Nevertheless existing 3D information should be used to align the model.

The initial pose of the matched 3D model provides a rough overall scale and position of the object, allowing us to identify with some accuracy which parts of the 3D reconstruction correspond to the model we wish to align.

We note $\mathcal{V}^i_{\text{visible}}$ the set of the visible 3D model vertices from camera i. When MVS reconstruction is available, we associate to a vertex v its closest point v_{mvs} on the line view. $E_{3D}(\Lambda)$ is defined as:

$$E_{3D}(\Lambda) = \sum_{i=1}^{n} \sum_{(v, v_{\text{mvs}}) \in \mathcal{C}^i} \|v - v_{\text{mvs}}\|^2 \quad (4)$$

with $\mathcal{C}^i$ the set of valid (v, v_{mvs}) pairs obtained from view i. As illustrated in Figure 6, when the reconstruction is far from the model, it is unlikely that v_{mvs} is part of the car and it should not be considered. In our case we use $1/5$ of the object length as the filtering threshold. This is a common strategy in point cloud alignment literature [45].

Constraints on the silhouette. For silhouette matching we first need to identify relevant contours in each image.

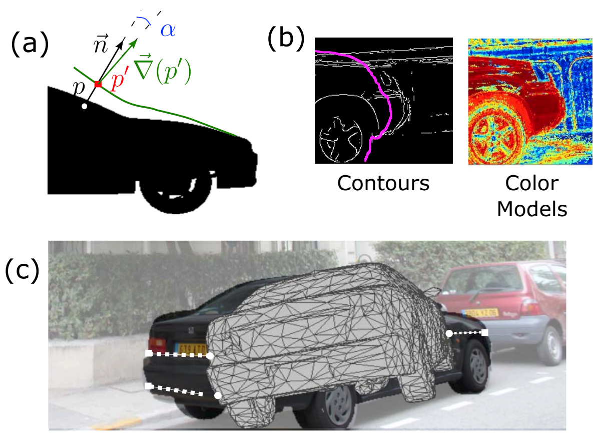

Figure 7. **Constraints from silhouettes.** (a) For a point p on the mesh contour we look for matching point along the line defined by $\vec{n}$. A candidate matching point p' is added as 2D constraint if $\alpha < 15°$. (b) In certain situations, the mesh contour in pink can have two corresponding contour points. In this case it is necessary to use color models (warm colors indicate high object probability). (c) The resulting constraints help align the 3D mesh (in gray).

We use the Canny filter [5] to detect edges in the images. The output is an edge map only based on local color differences. Isola *et al.* [24] describe a method to estimate the statistical dissociation information on boundaries. We use this information to filter the edge map and only keep edges likely to correspond to object contours. Once the contours in the image have been identified, the next step is to match points from the object's silhouette with edges detected in the image. If we consider p a point on the silhouette and $\vec{n}$ the normal to the silhouette at this point, we search for matching candidates along the normal line passing by p. We keep the two closest edge points to p as a matching candidate. This procedure can generate a large number of candidate matches with potentially many outliers. We first use the normals as filtering criteria. Let $\vec{\nabla}(p')$ be the color gradient at the image candidate pixel p'. If the angle difference between the vectors $\vec{\nabla}(p')$ and $\vec{n}$ is larger than $15°$, the candidate pixel p' is discarded (Fig. 7.(a)). This criterion is sufficient in many cases but certain situations require the usage of color information (Fig. 7.(b)). Pixels inside and outside the model projection are used respectively to estimate a foreground and a background color model. Here, these color models are histograms noted $\mathcal{H}^F$ and $\mathcal{H}^B$. Figure 7 shows the resulting *object* probability for pixels (warm colors indicate high object probability). We use these models to help filter incorrect matches by defining the *color* energy associated to a contour point p and a vector $\vec{n}$ as:

$$E_{\text{color}}(p,n) = \sum_{p \in \mathcal{S}_{\text{int}}} -log(\mathcal{H}^F(I_p)) + \sum_{p \in \mathcal{S}_{\text{ext}}} -log(\mathcal{H}^B(I_p)) \tag{5}$$

$\mathcal{S}_{\text{int}}$ and $\mathcal{S}_{\text{ext}}$ are set of points sampled along the line defined by $\vec{n}$. They respectively correspond to interior and exterior points. The defined energy is lower when interior and exterior points respectively satisfy foreground and background color distributions. Any silhouette match that results in an increase in contour energy is ignored. In essence, this is similar to energy terms used in level set segmentation [9].

We further enforce multi-view coherence by extending our filtering scheme. Each 2D constraint is transformed into a 3D constraint by using the vertex depth value. These 3D constraints are projected in all the other views, resulting in new 2D displacements. Views where this displacement causes an increase in the appearance energy E_{color}, vote to drop the constraint. We only keep the constraints for which a majority of views agrees. The energy term from silhouette matching is defined as:

$$E_{edge}(\Lambda) = \sum_{i=1}^{n} \sum_{(p,p') \in \mathcal{M}^i} ||p - p'||^2 \tag{6}$$

$\mathcal{M}^i$ is the set of 2D matching points from silhouettes in view i.

Optimization. We solve this alignment problem (Eq. 3) using gradient descent. Differentiation with respect to each pose parameter is provided in supplemental material. Figure 8 shows the result of the alignment step.

6. Geometry Morphing and Rendering

We now have a well aligned 3D model in our scene. However, due to the inevitable differences between 3D models retrieved and the 3D object observed, the 3D mesh needs to be adapted to fit the object in the images as best as possible. A key element of our approach is that we *automate* this process using the 2D silhouette matching obtained in the previous step, contrary to previous methods [26] based on manual annotation.

6.1. Mesh simplification

Mesh deformation techniques, such as the As Rigid As Possible (ARAP) morphing [49], require high quality manifold meshes. Meshes available in ShapeNet[6] often contain duplicate vertices and faces, self-intersecting polygons, disconnected components and they are generally unsuitable for further geometry processing.

We tried different methods to repair the meshes, including resampling the hull of the meshes with points and using screened Poisson reconstruction. While the resulting meshes are reasonable, small holes due to details in the 3D modelled mesh remained which create problems during rendering. As a reasonable compromise, we opted for the *semi-convex hull* representation [21]. This method preserves the shell of the objects, and outputs a manifold mesh suitable for further processing.

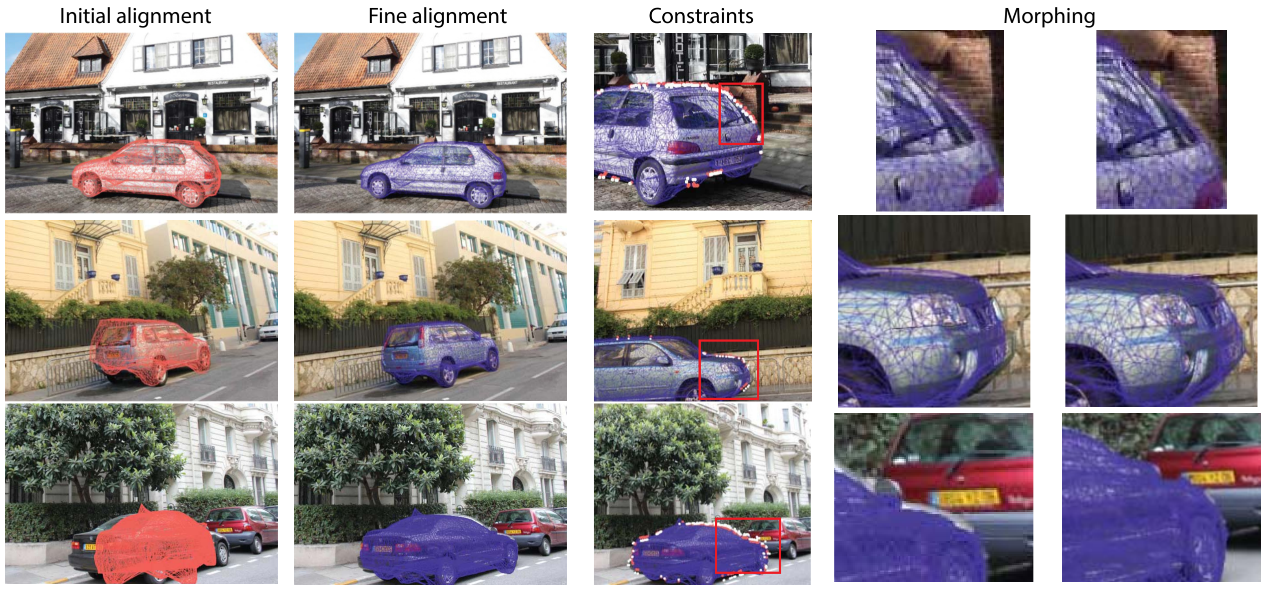

Figure 8. **Mesh alignment and morphing.** (a) Using 2D detection bounding boxes, an initial transformation of the 3D model is computed. (b) Using available 3d reconstruction and constraints from silhouette matching, a fine grain alignment of the model is estimated. (c) Because the 3d model does not always correspond to the images, deforming the mesh is necessary. Red points indicate mesh vertices to be displaced. White points indicate their target position. (d&e) After morphing, we obtain a better alignment of the 3d model.

6.2. Morphing

To deform the mesh, we obtain 3D constraints on vertices from 2D silhouette matching. For every silhouette point p matched with an edge point p' in the image, we obtain the new position of the mesh vertex v^i projecting on p. This new position, $v^i_{\mathcal{M}}$, is located along the viewing line of p' at the same depth as v^i. We enforce a smooth deformation of the rest of mesh using the As-Rigid-As-Possible surface deformation framework [49].

6.3. Rendering

Rendering proceeds in three passes. The first pass renders the background environment using the mask generated by the previous process. We use the algorithm of [38] to render this layer, and modify the shader to discard all pixels on the objects which have geometry using the mask. We see this layer in Fig. 9(left), which is the result of blending the four closest views. The second pass performs a ULR-like rendering of the car using the aligned and morphed geometry (Fig. 9-middle). Specifically we used deferred shading to render the depth and normals of the 3D model, look up the color in the closest images and use the ULR blending weights [4] to synthesize the final color on the object. Finally we blend background and object layers directly on the GPU to produce the final result (Fig. 9-right).

7. Results and Comparisons

We evaluate our method on 5 scenes. We use the scenes YellowHouse and Street from [7] that contain cars. We also propose the new scenes HotelBruges, Bosquet and Street2. Results of our rendering are shown in Figures (1, 9 and 10) but are best appreciated by watching the accompanying video[1] .

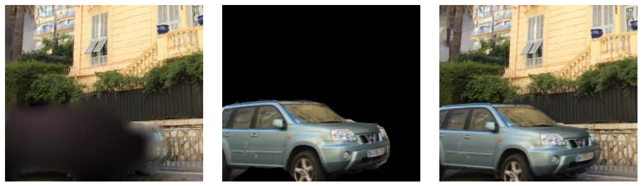

Figure 9. **Mixed-rendering.** Left: Background layer. Middle: Foreground object. Right: Final novel view.

Comparisons. We compare our method with state of the art rendering algorithms [4, 38]. To show the importance of the alignment step, we also compare with rendering based on the initial pose of the 3D model. Figure. 10 shows the result of rendering for each algorithm, at a novel viewpoint far from the input cameras. ULR [4] relies entirely on the available geometry and errors in the reconstruction are particularly visible (one can also notice the black regions where no 3D data exist). The selective approach using superpixels [38], performs better on the background in general. It compensates a little for errors in the geometry (1st row), but starts to show strong artifacts as we move closer to the background of the scene (3rd and 4th row). Just using the initial pose of the stock model may improve the rendering when no 3D data is available (row 3) but results are blurry and ar-

[1]http://team.inria.fr/graphdeco/publications

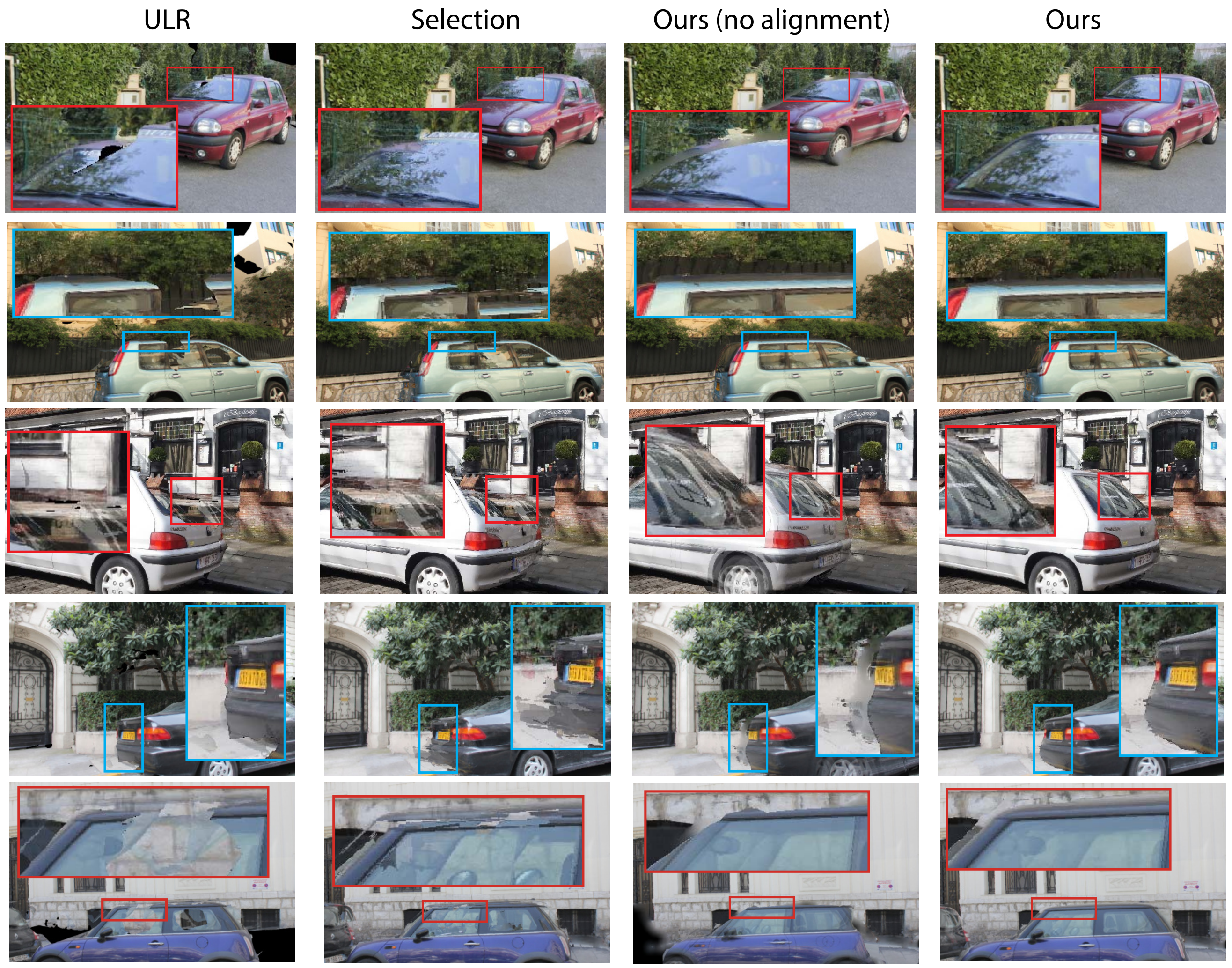

Figure 10. **Comparisons.** Rendering of novel viewpoints on Bosquet, YellowHouse, HotelBruges Street and Street2 datasets (from top to bottom). The rendering methods are, from left to right, ULR [4], Selection [38], Ours without alignment, Ours after alignment.

tifacts are created around edges. Our approach outperforms other rendering algorithms and it is able to compensate for the errors in the geometry. Thanks to alignment and morphing, the renderings around contours look natural and much fewer artifacts are visible. It now possible to move closer to objects in the scene contrary to previous methods.

8. Conclusion

In this paper, we introduce a new mixed Image-Based Rendering algorithm that builds on recent advances in object detection and recognition. We propose an entirely automatic pipeline that starts from object detection in images, then accurately places the object in the scene using a multi-view approach taking advantage of available geometry and silhouettes. As the stock 3D model may not exactly correspond, the 3D mesh is morphed to better fit the images. Our results show that we obtain improved rendering quality even when moving away from the input cameras. For the moment only car 3D model databases are rich enough to be used in our context, but our method is generic and can be directly applied on other object categories. We see our approach as a first step in a more general trend, in which traditional 3D models will be combined with image-based techniques to greatly simplify 3D content creation and interactive display.

Acknowledgments

Research funded by EU FP7 project 611089 CR-PLAY and French ANR project SEMAPOLIS (ANR-13-CORD-0003). We thank S. Bonopera for coding support.

References

[1] R. Arandjelović and A. Zisserman. Smooth object retrieval using a bag of boundaries. In *2011 International Conference*

on Computer Vision, pages 375–382. IEEE, 2011. 2

[2] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3762–3769, 2014. 2

[3] M. Aubry and B. C. Russell. Understanding deep features with computer-generated imagery. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2875–2883, 2015. 2

[4] C. Buehler, M. Bosse, L. McMillan, and S. G. M. Cohen. Unstructured lumigraph rendering. In *SIGGRAPH*, 2001. 2, 4, 7, 8

[5] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986. 6

[6] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. 2, 3, 4, 6

[7] G. Chaurasia, S. Duchene, O. Sorkine-Hornung, and G. Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. on Graphics (TOG)*, 2013. 1, 2, 7

[8] T. Chen, Z. Zhu, A. Shamir, S.-M. Hu, and D. Cohen-Or. 3-sweep: extracting editable objects from a single photo. *ACM Transactions on Graphics (TOG)*, 32(6):195, 2013. 3

[9] D. Cremers, M. Rousson, and R. Deriche. A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape. *International journal of computer vision*, 72(2):195–215, 2007. 6

[10] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, 2005. 2

[11] S. Dambreville, Y. Rathi, and A. Tannenbaum. A framework for image segmentation using shape models and kernel space shape priors. *IEEE transactions on pattern analysis and machine intelligence*, 30(8):1385–1399, 2008. 2

[12] S. Dambreville, R. Sandhu, A. Yezzi, and A. Tannenbaum. Robust 3d pose estimation and efficient 2d region-based segmentation from a 3d shape prior. In *European Conference on Computer Vision*, pages 169–182. Springer, 2008. 2

[13] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *SIGGRAPH*, 1996. 2

[14] J. M. Ferryman, A. D. Worrall, G. D. Sullivan, K. D. Baker, et al. A generic deformable model for vehicle recognition. In *BMVC*, volume 1, page 2. Citeseer, 1995. 2

[15] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. PAMI*, 2010. 5

[16] S. Gidaris and N. Komodakis. Object detection via a multiregion & semantic segmentation-aware CNN model. *CoRR*, abs/1505.01749, 2015. 2, 3, 4

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2

[18] M. Goesele, B. Curless, and S. M. Seitz. Multi-view stereo revisited. In *CVPR*, 2006. 1

[19] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections. In *ICCV*, 2007. 2

[20] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The Lumigraph. In *SIGGRAPH*, 1996. 2

[21] F. Guney and A. Geiger. Displets: Resolving stereo ambiguities using object knowledge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4165–4175, 2015. 6

[22] M. Hödlmoser, B. Micusik, M. Pollefeys, M.-Y. Liu, and M. Kampel. Model-based vehicle pose estimation and tracking in videos using random forests. In *2013 International Conference on 3D Vision-3DV 2013*, pages 430–437. IEEE, 2013. 2

[23] D. P. Huttenlocher and S. Ullman. Object recognition using alignment. In *ICCV*, 1987. 2

[24] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson. Crisp boundary detection using pointwise mutual information. In *ECCV*, 2014. 6

[25] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3121–3128. IEEE, 2011. 1, 2, 3, 5

[26] N. Kholgade, T. Simon, A. Efros, and Y. Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Transactions on Graphics (TOG)*, 33(4):127, 2014. 1, 3, 6

[27] D. Koller. Moving object recognition and classification based on recursive shape parameter estimation. In *Proc. 12th Israel Conf. Artificial Intelligence, Computer Vision*, volume 2728, 1993. 2

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 4

[29] J. Lee, Y. Kim, S. Lee, B. Kim, and J. Noh. High-quality depth estimation using an exemplar 3d model for stereo conversion. *IEEE transactions on visualization and computer graphics*, 21(7):835–847, 2015. 1, 3

[30] M. J. Leotta and J. L. Mundy. Vehicle surveillance with a generic, adaptive, 3d vehicle model. *IEEE transactions on pattern analysis and machine intelligence*, 33(7):1457–1469, 2011. 2

[31] M. Levoy and P. Hanrahan. Light field rendering. In *SIGGRAPH*, 1996. 2

[32] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing IKEA objects: Fine pose estimation. In *ICCV*, 2013. 2

[33] Y.-L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *European Conference on Computer Vision*, pages 466–480. Springer, 2014. 2

[34] C. Lipski, F. Klose, and M. Magnor. Correspondence and depth-image based rendering: a hybrid approach for free-viewpoint video. *IEEE T-CSVT*, 2014. 1, 2

[35] D. Lowe. The viewpoint consistency constraint. *IJCV*, 1(1):57–72, 1987. 2

[36] F. Massa, B. C. Russell, and M. Aubry. Deep exemplar 2d-3d detection by adapting from real to rendered views. *CoRR*, abs/1512.02497, 2015. 2, 4

[37] R. Mottaghi, Y. Xiang, and S. Savarese. A coarse-to-fine model for 3d pose estimation and sub-category recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 418–426. IEEE, 2015. 2

[38] R. Ortiz-Cayon, A. Djelouah, and G. Drettakis. A Bayesian Approach for Selective Image-Based Rendering using Superpixels. In *International Conference on 3D Vision - 3DV*, Lyon, France, Oct. 2015. 2, 3, 7, 8

[39] M. Prasad and A. Fitzgibbon. Single view reconstruction of curved surfaces. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1345–1354. IEEE, 2006. 3

[40] V. A. Prisacariu and I. Reid. Nonlinear shape manifolds as shape priors in level set segmentation and tracking. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2185–2192. IEEE, 2011. 2

[41] V. A. Prisacariu and I. D. Reid. Pwp3d: Real-time segmentation and tracking of 3d objects. *International journal of computer vision*, 98(3):335–354, 2012. 2

[42] V. A. Prisacariu, A. V. Segal, and I. Reid. Simultaneous monocular 2d segmentation, 3d pose recovery and 3d reconstruction. In *Asian Conference on Computer Vision*, pages 593–606. Springer, 2012. 3

[43] L. Roberts. Machine perception of 3-D solids. In *PhD. Thesis*, 1965. 2

[44] B. Rosenhahn, T. Brox, and J. Weickert. Three-dimensional shape knowledge for joint image segmentation and pose tracking. *International Journal of Computer Vision*, 73(3):243–262, 2007. 2

[45] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001. 5

[46] R. Sandhu, S. Dambreville, A. Yezzi, and A. Tannenbaum. A nonrigid kernel-based framework for 2d-3d pose estimation and 2d image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(6):1098–1115, 2011. 3

[47] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 2, 4

[48] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM Trans. Graph.*, 2006. 1, 2

[49] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, 2007. 6, 7

[50] H. Su, Q. Huang, N. Mitra, Y. Li, and L. Guibas. Estimating image depth using shape collections. *ACM Transactions on Graphics (Proceeding of SIGGRAPH)*, 33(4), 2014. 2

[51] C. Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE, 2013. 3

[52] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3057–3064, June 2011. 3

[53] K. Xu, H. Zheng, H. Zhang, D. Cohen-Or, L. Liu, and Y. Xiong. Photo-inspired model-driven 3d object modeling. In *ACM Transactions on Graphics (TOG)*, volume 30, page 80. ACM, 2011. 3

[54] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014. 2

[55] Y. Zheng, X. Chen, M.-M. Cheng, K. Zhou, S.-M. Hu, and N. J. Mitra. Interactive images: cuboid proxies for smart image manipulation. *ACM Trans. Graph.*, 31(4):99–1, 2012. 3

[56] M. Zhu, X. Zhou, and K. Daniilidis. Single image pop-up from discriminatively learned parts. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 927–935, 2015. 2

[57] C. L. Zitnick and S. B. Kang. Stereo for image-based rendering using image over-segmentation. *IJCV*, 2007. 1