3D RECONSTRUCTION FROM A SINGLE IMAGE

By

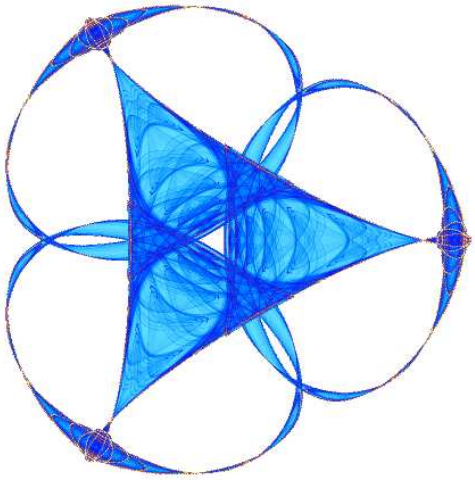
Diego Rother

and

Guillermo Sapiro

IMA Preprint Series # 2221

(August 2008)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA

400 Lind Hall

207 Church Street S.E.

Minneapolis, Minnesota 55455–0436

Phone: 612-624-6066 Fax: 612-626-7370

URL: http://www.ima.umn.edu

3D Reconstruction from a Single Image

Diego Rother and Guillermo Sapiro

Abstract— A probabilistic framework for 3D object reconstruction from a single image is introduced in this work. First, a probabilistic generative model to represent the distribution of mass of a class of objects in 3D space, namely a 3D shape prior, is presented. Next, following the Beer-Lambert law in optics, a framework to translate these 3D probabilities into the corresponding 2D probabilities in the camera plane is developed. Exploiting this framework to encode prior knowledge about the class and to project it to 2D, the problem of 3D reconstruction from a single image is casted as a statistical inference problem in graphical models, where actual observations in the single image are naturally integrated with 3D prior knowledge of the class. The reconstruction is obtained by running *modified* belief propagation in this graphical model, and in some cases, optimal solutions are guaranteed. The proposed modification allows the exact computation of the messages to pass in quasi-linear time, a significant improvement over the exponential time complexity of general implementations. The presentation of the proposed framework is complemented with evaluation of the experimental results obtained for the important class of "walking people," demonstrating the accuracy of the approach for 3D reconstruction, localization and volume estimation.

Index Terms— Belief propagation, graphical models, 3D reconstruction, single image scene analysis, volumetric statistical image representations.

1 Introduction

It is common for human subjects to perceive three dimensional (3D) objects when presented with two dimensional (2D) images alone, by relying on prior knowledge about the world and the objects involved. Machines, on the other hand, still perform poorly on this task. This paper addresses this problem of 3D reconstruction of an opaque object, from a single image, by exploiting prior knowledge about the class of the object being reconstructed. This prior knowledge is encoded in what we call a *3D shape prior* (or 3D prior, for short): a structure that encodes the probability that a given portion of 3D space, relative to a reference position on the ground, is occupied by an object of the given class, when an object of that class is placed at the reference position. The 3D reconstruction is the result of the interplay between the prior knowledge and the background probability computed from a single input image using a background model. No other cues from the image, such as colors, self-shading, or texture gradients, are used.

Figure 1 illustrates the intuition behind 3D priors (a formal description is postponed until Section 3). Four members of the class "vases" have been observed. The 3D prior for this class is learned by: 1) defining a bounding box and discretizing it into voxels; 2) placing this box at the position where each object is standing; [1] and 3) computing, for each voxel, the fraction of observed instances of the class that contain the voxel. These voxel fractions, or occupancy probabilities, constitute the 3D shape prior, a *generative* model for the objects of the given class.

3D priors can be used to represent classes of diverse solids.[2] Variation within a class can be produced by the evolution of a solid's shape (as in the case of a person walking) and/or simply by the inclusion in the class of different rigid objects (as the vases in Figure 1). In this article, the exposition and examples are restricted to the important class of "walking people," however other classes can be handled analogously; the 3D prior is what changes, the proposed probabilistic inference framework remains the same.

Armed with the 3D prior of a class, we can answer questions of the kind: "if an object of the class is standing at position x, what is the probability that voxel y (defined with respect to x) will be occupied by the object?" This is the kind of critical question that needs to be addressed in common reconstruction problems, when such reconstruction cannot be carried out using the information provided by the actual observations alone. It is also possible, with this prior, to answer another related question: "given that the set of pixels Z in an image are known to be occupied, what is the probability that an object of the studied class is standing at x?" This kind of question often arises in the context of model-based tracking and localization, and is related to the quality of the fit between the class model and the observations. Our overall goal in this work is to "understand" the observed 3D space with sufficient accuracy to address important tasks such as scene learning (as in [1]), pose estimation, localization, navigation, counting, and model-based tracking, in addition to 3D reconstruction. The application of 3D priors to address some of these problems is described in Section 5.

Using the 3D priors formulation, the reconstruction problem is framed as a statistical inference problem in *graphical models*. A graphical model that relates the (3D) voxel states (the output of the system), the pixel observations (the input), and the 3D prior (the prior knowledge), is defined to make inferences about the 3D structure of the unknown object. In doing so, a law necessary to project 3D probabilities into the 2D image plane, and hav-

[1] Throughout this article, to "place the box at a position x" means to "set the box in the scene, resting on the floor, with the center of its lower face on top of position x." Similarly, when an "object is standing at x," we mean that "its center of mass is vertically above x."

[2] The definitions for the word "solid" relevant in this context are: 1) a three dimensional shape; and 2) having three dimensions.

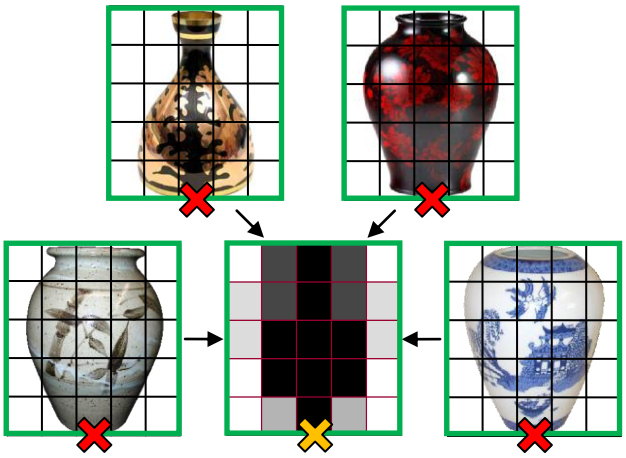


Figure 1: The idea of the 3D shape prior, here represented in 2D for illustration purposes. If a new vase (whose shape is unknown) is located on the yellow cross, what is the probability that a given voxel will be occupied by it? This probability constitutes the 3D (2D in the figure) prior and is represented in grayscale (white is 0, black is 1) in the middle bounding box. This prior is computed from the sample of known vases shown surrounding the middle box. The bounding boxes (in green) enclose both the prior and the samples. The position of all voxels in a box is defined relative to the reference position of the box (indicated with the cross).

ing the desired mathematical properties, is postulated by analogy with the Beer-Lambert law in optics. The graphical model defined constitutes a natural framework to integrate prior 3D information with actual 2D observations (the image). Inference is very efficiently performed by running a modified instance of belief propagation, and in some cases, optimal solutions are guaranteed. The proposed modifications allow, in this case, the computation of the messages to pass in quasi-linear time, a significant improvement over the exponential time complexity of general implementations.

The remainder of this paper is organized as follows. Section 2 places the current work in the context of prior relevant work, discussing important connections. Sections 3 and 4 introduce the components of the proposed graphical model, and the efficient algorithms to make inferences on it, respectively. Section 5 presents experimental results obtained with the proposed framework, and Section 6 concludes with a discussion of the key contributions and directions for future research.

2 Prior Work

Three dimensional reconstruction from images, understood in the broadest sense, is arguably one of the most studied problems in the area of computer vision, as attested by the diversity of names in use to refer to different instances of this problem, and by the vast number of publications addressing them. To organize (part of) this corpus, here we look at three dimensions of the problem: 1) the amount/type of input received; 2) the generality of the output representation; and 3) the complexity of the prior information available.

The amount of input information received depends (mainly) on the number of input views. Different instances of the problem range from multi- to single-view reconstruction. *Multi-view reconstruction* (e.g., [2], [3], [4], [5]) uses information from two or more views (often significantly more than two) of the object or scene, while *single-view reconstruction* (e.g., [6], [7], [8], [9], [10]), at the opposite end, uses information from only a single view. *Stereo reconstruction* (reviewed in [11]) is a special case of multi-view reconstruction, which is sufficiently important to deserve a name of its own. In this case, two cameras, often placed close together with respect to the object, are used.

The second dimension of the 3D reconstruction problem is the generality of the objects that can be represented. This dimension ranges from representations that are general enough to characterize any object (*general representations*), to specific representations that are better suited to represent objects from one particular class (*specific representations*). There are two main general representations: volumetric and surface based. *Volumetric representations* typically discretize a volume containing the whole object into voxels, and estimate the occupancy of the voxels that best fit the input image/s according to some metric (see for example [2], [3] for a review). Surface-based representations (see for example [4], [5], and references therein) build a mesh (or point cloud or parametric) representation of the object's visual hull by computing the intersection of the viewing cones associated with the object's observed silhouettes.

Specific representations are particular to one class of objects, and many have been defined in the context of model-based tracking, or pose estimation, in addition to reconstruction. For the class of "walking people" in particular, many models have been proposed (see [12] for a survey), such as articulated bodies ([7], [8], [9], [10], [13]), generalized cylinders [14], silhouettes ([15], [16]), and models capable of producing (at a higher computational cost) visually appealing reconstructions ([17], [18]). Some of these models are not limited to representing "walking people," but can represent other classes (e.g., "horses" as shown in [7]). Articulated bodies, modeling the different degrees of freedom of the human body components, are among the most popular 3D representation for the class of "walking people" [12].

By limiting what can actually be represented, specific representations condition the reconstruction. If the object being reconstructed can be properly encoded in the specific representation, in general it can be reconstructed with less actual input (i.e., fewer images). However, if the object cannot be accurately handled by the specific representation (e.g., trying to reconstruct a mountain with an articulated model), then evidently the output of the reconstruction will not be an accurate representation of the object (it simply cannot be), and often results in hallucinations (the model is found where it is not present). In contrast, general representations able to represent any object at any desired (but predefined) approximation quality, are less informative, and in general require more information in order to reconstruct a known class object with the same quality. A clear relationship exists between these two dimensions defined above. Algorithms receiving in-

put from several views do not necessarily need to rely on *a priori* information during the reconstruction, and therefore are able to reconstruct more general objects. In general, multi-view reconstruction is solved using general representations (volumetric or surface based), and no or little prior knowledge (as for example in [2], [3], [4], [5], [19], [20], [21]). On the other hand, specific models are used to reconstruct objects when only one or two views are available, which by themselves do not provide enough information for a good reconstruction (e.g., [6], [7], [8], [9], [10], [15], [14]).

In addition to (or instead of) constraining the set of objects that can be actually reconstructed, a probability density (or energy) can be defined on the space of objects that can be represented. This probability density, also known as a *prior*, can be used to discriminate between two feasible reconstructions that explain the input when one is known to be more likely than the other. This prior is the third dimension we use to classify reconstruction approaches. It is related to the generality of the reconstruction, but it is less "drastic" in the sense that reconstructions are not completely forbidden but rather discouraged. Priors have been designed, for example, to reward smoothness in volumetric reconstructions [19], planarity of faces and similarity of angles and lengths in surface-based reconstructions (to reconstruct polyhedra) [6], smoothness of evenly spreading curves in space (to reconstruct grasses) [6], and viability of joint angles and movements in articulated models [9]. An advantage of defining a prior on the reconstructions, compared with the strategy of constraining the representation space (as in specific representations), is that this prior can be automatically learned from training data for particular classes, without having to "redesign" the representation for each particular new class.

The framework proposed in this work can now be placed in the coordinate system just defined. A *general volumetric representation*, generally used in multi-view settings, is used in this case to reconstruct a 3D volume from a *single* view. This volumetric representation is augmented by defining a (3D) prior on the possible reconstructions, learned from training data for a particular class. In this coordinate system, the work by Snow *et al*. [19] is perhaps the closest, since it uses a volumetric representation and a prior on possible reconstructions. However the prior used by Snow *et al.* simply rewards smoothness and has no further information about the object class itself, as 3D priors do. An additional difference is that Snow *et al.* use data from multiple views. Thereby, to the best of our knowledge, our work is the first one that is located in this important corner of this "three dimensional cube" defined by the amount/type of input, the generality of the 3D representation, and the richness of the reconstruction prior.

The 3D prior framework here introduced is related to contour-based approaches (e.g., [22]), in the sense that it only relies on the distribution of mass in space and not on the appearance "inside the silhouette."[3] Therefore, as those approaches, it is less sensitive to camouflage or changes in color produced by lighting or shadows. For this reason, they are complementary to other 3D object models that rely on the appearance "inside the silhouette" (e.g., [23] and [24]). Unlike contour-based models however, 3D priors "live" in 3D (i.e., points in the 3D prior have a specific corresponding 3D position in the world), and therefore can be used for 3D reconstruction, integrating camera matrix or other 3D information (e.g., the constraint that objects do not occupy the same space at the same time).

Some articulated body models do "live" in 3D (e.g., [8], [9], [10]), and therefore can be used to recover simple 3D reconstructions from a single view (sometimes referred as *pose estimation*). In contrast with our approach, these reconstructions are mainly skeletons, thin or thick, and cannot (and do not intend to) naturally address the reconstruction of clothes, backpacks, or other accessories. Articulated bodies perform pose estimation by solving the assignment problem, i.e., by establishing which pixels correspond to what body parts. However, if only the 3D reconstruction is required (not the assignment of body parts to pixels), the 3D priors framework provides this information without explicitly solving the assignment problem. This is a significant conceptual difference between the two frameworks.

The present work is possibly most related to the work by Franco and Boyer [20], since both define a probabilistic graphical model that relates an occupancy grid to the observed images. However their work differs from ours in a number of very significant aspects: 1) Franco and Boyer use *multi-view* information, while we use information from a *single* view/camera; 2) they use a uniform prior, while we define a (learned) class specific 3D prior, necessary to provide the information not available in the single view; 3) the projection model used to compute a probability in the 2D image from the 3D occupancy probability is radically different (see Section 3.3); and 4) the inference algorithms, acting on very different graphical models, are significantly different (see Section 4.2).

After introducing the proposed framework, in Section 4.3 we further discuss connections with other prior work.

3 The Graphical Model

This section presents the key components and variables of the proposed 3D inference framework, together with the proposed statistical graphical model that describes the relationships among them. A representative part of this graphical model is shown as a *factor graph* in Figure 2. Factor graphs, [25], are graphical constructions to represent the product of functions of several variables. There are two kinds of entities in a factor graph: variable nodes (represented with circles in the figure), and factor nodes (represented with squares in the figure). Each variable node corresponds to a variable in the system, and each factor node corresponds to a factor in the product of functions. A connection is established between a variable node and a factor node if and only if the corresponding factor function contains the corresponding variable. Con-

[3] Except to distinguish foreground from background.

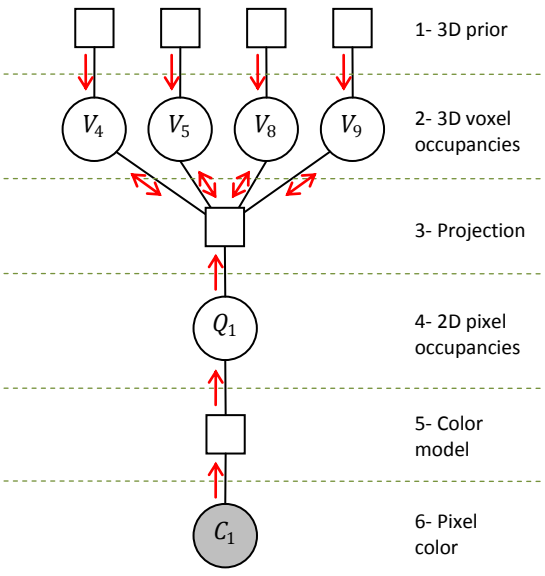


Figure 2: Factor graph for a visual ray, its corresponding pixel, and the voxels it intersects. A factor graph, [25], has a *variable node* (represented as a circle) for each variable, and a *factor node* (represented as a square) for each factor in the joint probability expression of the system. Factor nodes are connected to the variable nodes corresponding to the variables that appear in the factor. Red arrows indicate the direction of the messages passed along each link. Observed variables are shaded.

nections are not allowed between variable nodes or between factor nodes; factor graphs are bipartite.

There are three kinds of variables in our system (Figure 2): the input variables that are the colors (or appearance in general) of the pixels in the single input image (in level 6), the output variables that are the states ("empty or "occupied") of the voxels (in level 2), and the hidden variables that represent the states of the pixels (in level 4).[4] In addition to these core variables, there are in our system three kinds of factors, defining the probabilities of the variables immediately "below" them, possibly conditioned on the variables right "above" them.

Causality is depicted flowing downwards in the graph, in other words, the color of a pixel depends on the state of the pixel, which in turn depends on the states of the voxels "above" it. On the other hand, current information flows upwards (red arrows in Figure 2), from the input (the pixel colors) to the output (the voxel states), where it is integrated with the prior knowledge flowing downwards. The following sections describe each of the levels in this graph in more detail.

3.1 The 3D Shape Priors

As mentioned above, data from a single view is not enough to recover full 3D structure, and it has to be supplemented, e.g., with prior knowledge.[5] In this work, the prior knowledge is encoded in the entity that we call a *3D Shape Prior*. In this section we extend the models that were introduced in [1] to provide this needed extra information, leading to the reconstruction of an object of a known class from a single view.

As mentioned in Section 1 and illustrated in Figure 1, 3D priors keep track of the distribution of mass in space, with respect to a reference position in the ground, χ, for the members of a given class. It is assumed throughout this work that all objects of the class are completely opaque, i.e., the background cannot be seen through any part of the objects. In addition, we only consider bounded/finite objects. Therefore, there exists an (imaginary) 3D *bounding box* large enough to contain each one of the objects in the class. One last assumption about the objects in the class is that they are characterized by a single "size" parameter, ρ. In other words, this means that the length/height and width/height ratios of all the objects in the class are *approximately* constant; their size, however, is variable. The validity of this assumption evidently depends on the specifics of the class and is partially accurate for the class of objects that we test in this article (i.e., walking people). Nevertheless, this first order approximation works reasonably well and is not fundamentally indispensable in the proposed framework; additional "size" parameters can be added without substantively altering the framework.

Note in Figure 1 that the bounding boxes are all registered with the objects they contain, standing in the same place (indicated with a cross in the figure), and having roughly the same height. In 3D, registration requires fixing (or estimating) an additional parameter, namely the rotation of the bounding box around the (vertical) z-axis, θ. As will become clear in Section 3.2, the registration parameters (χ, ρ, and θ) that align all the sample objects of the class with the bounding box are a necessary input to correctly learn the prior. In Section 5.4 we describe how, having already learned the prior, these parameters will be unknown and will be automatically estimated as part of the reconstruction process.

The bounding box is divided into equally sized cubes, named *voxels* (Figure 1). We model these voxels as independent Bernoulli random variables, V_i, with two possible occupancy states: *empty* (0) or *occupied* (1). These variables are depicted in level 2 (Figure 2).

The parameters associated with the distributions of these Bernoulli random variables (their *success rate,* or *a priori occupancy probability* in this context), are not uniform and have to be learned (this is explained in Section 3.2). $P(V_i = 1)$ is the probability that the voxel is occupied, and $P(V_i = 0) = \big(1 - P(V_i = 1)\big)$ is the probability that it is empty. The occupancy probabilities (or just "occupancies", for short) of voxels in areas of the bounding box that are consistently part of the objects of the class (e.g., voxels close to the vertical at the center of the bounding boxes in Figure 1), have values close to 1; occupancies of voxels that are never, or very infrequently, occupied by an instance of the class (e.g., voxels close to the left and right borders of the bounding boxes in Figure 1), have values close to zero (this resembles the concept of "matting" [26]). The 3D prior (in level 1) is a three dimensional

[4] All subsequent mentions of "levels" refer to the levels in Figure 2.

[5] In some special cases, like flat objects or solids of revolution, it is indeed possible to reconstruct an object from just a single view. In these cases, the knowledge of the object's class membership, i.e., knowing that the object belongs to the class of "flat objects" or "solids of revolution," constitutes the prior knowledge and is essential for its reconstruction.

array (or tensor), the same size (in voxels) as the bounding box, in which each element contains the *a priori* occupancy of the corresponding voxel Bernoulli variable.

This leads to a generative model of order zero that assigns the probability,

$$P(V_1, \ldots, V_M) = \prod_{i=1}^{M} P(V_i), \qquad (1)$$

to each of the 2^M possible voxel configurations (solids) that can be "generated" by the model. There is one factor in (1) for each voxel variable, and therefore each variable node in level 2 is connected to a single factor node in level 1 (again, see Figure 2).

The assumption of voxel independence, that allows to factorize (1), also considerably reduces the size of the model and, by avoiding loops in the resulting graphical model, it simplifies the inference (more on this in Section 4.1). It is possible to relax this assumption without introducing loops in the graph, by splitting a class into subclasses (e.g., the class of "walking people" can be split into the subclasses "right leg in front," "left leg up," "left leg in front," and "right leg up"). By doing so, voxels become only conditionally independent on each other (conditioned on the subclass label), while important dependencies between them are introduced (this is exploited in the experiments reported in Section 5.4). In the case of the class "walking people," the introduced dependencies provide information of the kind "if the right foot is in front, then the whole right leg must be in front." The price to pay for the additional complexity is an increase in the computational time and the number of model parameters, proportional to the number of subclasses.

We now describe how to learn this zero order 3D prior.

3.2 Learning the 3D priors

In [1], we introduced a simple and inexpensive method to accurately learn 3D priors using a single camera and the Radon transform. While we could certainly use this method in the work here presented (the proposed 3D inference framework is of course independent of how the 3D prior was actually learned), we use an alternative approach described next.

To learn the 3D priors we can use a multi-view reconstruction algorithm, in a multi-camera setup, to obtain solids that are then registered and averaged to obtain the 3D prior (this multi-view/camera is only done off-line for learning the 3D priors, our inference framework uses a single view/image at the time of reconstruction). A basic multi-camera method has the following key steps. First, a collection of solids are acquired in the multi-camera setup (e.g., as in [19]). Then, the solids are registered with respect to the bounding box. Hence, as mentioned in Section 3.1, the position χ, size ρ, and orientation θ of each solid must be determined. The position and size of a solid are easily recovered as the vertical projection in the floor of the solid's center of mass and its height, respectively. The orientation can be obtained in a number of ways, depending on the specifics of the class. One alternative is to compute for each solid the orientation that maximizes the intersection with a reference solid in the sample (e.g., the first one). A different alternative is presented in Section 5.1 for the particular class of "walking people." Next, each voxel in the bounding box (now registered with the solid) is given the label 0 or 1, depending on the fraction of it inside the solid. Finally, the 3D prior is obtained as the average, for each voxel, of the labels observed in the voxel.

In this method, since 3D reconstruction precedes time averaging, synchronization between views is essential, making this approach more complex and expensive than the one introduced in [1]. On the other hand, having a 3D reconstruction for each time can be used to learn higher order priors (i.e., priors whose voxels are not independent) or temporal priors (i.e., priors whose voxels depend on the state of voxels at different times). By contrast, in the single-camera method, the information necessary to recover these higher order priors is lost.

Since datasets containing 3D reconstructions for the class "walking people" are available [27], we adopted the multi-camera method, skipping the first step (Section 5.1).

3.3 Projecting 3D priors

We now describe the "image formation" process when 3D priors are involved, in other words, how the 3D occupancies associated with voxels in space, are projected to yield the 2D occupancies associated with pixels in the image plane (Figure 3). The occupancy state of pixel j (in level 4), denoted as Q_j, is a random variable that can be in one of two possible states: "empty" or "background" ($Q_j = 0$), or "occupied" or "foreground" ($Q_j = 1$). In this section we derive an expression for the occupancy of a pixel, $P(Q_j | V_1, \ldots, V_n)$, given the occupancy states, $V_1, \ldots, V_n$, of the voxels in its *line of sight*.[6]

There are a number of issues that the "image formation" process has to be able to handle:

1 When a voxel is partially occupied by an object, two rays traversing this voxel may end up with different outcomes, i.e., one ray may "see" it *empty* while the other sees it *occupied*. This phenomenon is most common in the voxels along the boundary of the object (e.g., in voxels along the borders of the vases in Figure 1).

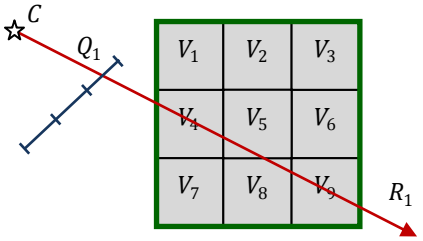


Figure 3: 2D analog of the projection process. A ray R_1, originating at the camera center C, passes through its corresponding pixel Q_1 in the image plane (in blue) and intersects the voxels in its line of sight (V_4, V_5, V_8 and V_9) inside the 2D bounding box (in green). The relationship among these *particular* variables is depicted in the factor graph of Figure 2.

[6] Voxels in the line of sight of a pixel are all those voxels that are intersected by the ray passing through the camera center and the center of the pixel. The terms "line of sight" and "ray" are used interchangeably. Since there is exactly one line of sight (or ray) per pixel, we often refer to the pixel and the voxels in *its* line of sight, or simply, *its* voxels.

2 Noise in the camera sensors or mistakes in the background probability computation (to be explained in Section 3.4) may disguise the state of a background pixel as foreground (or vice versa), wrongly suggesting that the voxels in the pixel's line of sight must be occupied.

These facts suggest that a stochastic (rather than a deterministic) relation exists between a voxel and the pixels where it is seen. To take these facts into account we define two constants, ε_0 and ε_1 ($0 < \varepsilon_0, \varepsilon_1 \ll 1$), such that a pixel with a single voxel in its line of sight has a low occupancy, ε_0, if the voxel is empty, and a high occupancy, $(1-\varepsilon_1)$, otherwise. It is important to note that both constants are *strictly* greater than zero, so that the state of the pixel cannot be certain in either state of the voxel. Section 5.2 presents a criterion to select values for ε_0 and ε_1 based on empirical considerations.

An intuitive interpretation of these constants that will turn out to be useful later is to consider that each voxel can be made of two different "materials" having different "optical" properties. [7] An *opaque* material fills the occupied voxels and "blocks light" with probability $(1-\varepsilon_1)$, and a *translucent* material fills the empty voxels and "blocks light" with probability ε_0. According to this interpretation, the 3D prior determines the probability of getting "opaque" when drawing samples from each voxel.

In this model then, the complement of the occupancy of a pixel Q_j, given the states of the voxels $V_1, \cdots, V_n$ in its line of sight, $P(Q_j = 0 | V_1, \cdots, V_n)$, is equivalent to the probability that "light," emitted from the camera center C and traversing through the pixel and its voxels, reaches the background (see Figure 3),

$$P(Q_j = 0 | V_1, \cdots, V_n) = \prod_{i=1}^{n} \mathrm{E}(V_i), \qquad (2)$$

where $\mathrm{E}(0) = 1-\varepsilon_0$ and $\mathrm{E}(1) = \varepsilon_1$.

The simple expression in (2) for the pixel occupancy is inadequate in (at least) two respects. First, the probability that a ray is blocked by a voxel must depend on the length traveled by the ray inside the voxel, so that the voxel has a small effect on rays that it slightly intersects and a larger effect on rays that it considerably intersects. This is critical to resolve ambiguities (break ties) when the outcomes of two rays that intersect the same voxel are different.

Second, in this expression the pixel's occupancy depends on the resolution of the bounding box chosen by the user, when in fact it should be independent of this arbitrary choice. Consider two different resolutions of the bounding box (but the same actual bounding box size): 2x2x2 and 1x1x1 voxels. If the box is completely full (i.e., all voxels are "occupied"), in the first resolution the (pixel) occupancy computed (using (2)) along a ray that intersect two voxels is $(1-{\varepsilon_1}^2)$, whereas, in the second resolution the occupancy computed along the same ray is $(1-\varepsilon_1)$. This discrepancy proves that the "image" of the box under this image formation process is not invariant to the box's resolution, and furthermore, it is arbitrary.

To address these problems we propose a new image formation formulation that, by exploiting the interpretation of priors as "material", borrows from the Beer-Lambert law used in optics. The *Beer-Lambert law,* [28], relates the absorption of light by a slab of material to the thickness of the slab and the optical properties of the material. Consider a slab of thickness dr of a material characterized by the absorption constant α. Assuming that the number dI of photons absorbed by the slab, per thickness dr, is proportional to both the number I of incident photons on the slab and the constant α of the material, it holds that

$$\frac{dI}{dr} = -I.\alpha.$$

Rearrangement and integration subject to the boundary condition $I = I_0$ at $r = 0$, yields the Beer-Lambert law,

$$\ln\frac{I}{I_0} = -\alpha.r \Rightarrow P = \frac{I}{I_0} = e^{-\alpha.r}, \qquad (3)$$

where P is the probability that a photon passes through the material and r is the distance traveled by the photon inside the material. We make the opaque and translucent materials defined above behave according to the Beer-Lambert law, with different absorption constants. Let the translucent material's constant be $\alpha_0 \triangleq -\ln(1-\varepsilon_0)$, so that the probability that a light ray passes through a translucent material slab of unit thickness is $e^{-\alpha_0} = 1-\varepsilon_0 \approx 1$, and let the opaque material's constant be $\alpha_1 \triangleq -\ln(\varepsilon_1)$, so that the probability that a light ray passes through an opaque material slab of unit thickness is $e^{-\alpha_1} = \varepsilon_1 \approx 0$.

From (3), it follows that the new relationship between pixel Q_j's occupancy and the occupancy of the voxels in its line of sight is,

$$P(Q_j = 0 | V_1, \cdots, V_n) = \prod_{i=1}^{n} e^{-\alpha(V_i).r_{ji}} = \prod_{i=1}^{n} \mathrm{E}(V_i)^{r_{ji}}, \qquad (4)$$

where $\alpha(V_i)$ is the absorption constant corresponding to voxel V_i ($\alpha(0) = \alpha_0$, and $\alpha(1) = \alpha_1$), and r_{ji} is the length traveled by the ray R_j inside of voxel V_i. These distances depend on the camera matrix considered, the position of the box, and its resolution (i.e., the number of voxels in the box along each dimension).[8] Equation (4) defines the relationship between the variables connected to the factor node in level 3 (the voxel occupancies above and the pixel occupancies below). This is the "image formation" law that we adopt, while Franco and Boyer, [20], adopted a law closer to (2).

Let us consider again the case of the same ray intersecting the same box at two different resolutions. Let us assume that the ray intersects two voxels in the first resolution (with intersection lengths ℓ_1 and ℓ_2 inside the voxels), and a single voxels in the second resolution (with

[7] Bear in mind that we are not dealing with the optical properties of the objects being reconstructed, these objects are completely opaque. Rather, the properties of the 3D priors for the class of the objects, are being considered.

[8] It is still necessary to use the resolution of the box in the calculations, but the answer (the "image" of the box) is independent of it.

length $\ell = \ell_1 + \ell_2$, since the ray and the boxes are the same). The occupancy computed according to (4) in the first resolution is now $\left(1 - e^{-\alpha_1 \cdot (\ell_1 + \ell_2)}\right)$, which is equal to that obtained in the second resolution. This example demonstrates that with this image formation process, (4), it is now possible to change the resolution of the prior (e.g., to obtain a reconstruction at a higher resolution), without affecting the "image" of the box. Furthermore, it is even possible to mix voxels of different sizes (e.g., using octrees *a la* [29]) to improve or accelerate the reconstruction. The "optical properties" are now intrinsic to each *point* in space and not to each *voxel* (as they are if using the more standard model (2)), permitting not only changes in resolution, but also in the box's size, and even in its geometry. The importance of this is further exposed in Section 4.1. The increase in the computational complexity of (4) vs. (2), on the other hand, is negligible.

3.4 Color Models

A pixel occupancy, Q_j, is not directly observed, rather what is observed is the color (or attributes in general) in its corresponding pixel, C_j (in level 6). These pixel colors in the single input image are the only observable variables in the system, and the actual pixel occupancies have to be inferred from them.

Pixel colors are considered continuous random variables in the RGB color space. The probability density of the colors expected at the pixel C_j depend on the pixel's occupancy state Q_j. If $Q_j = 0$ (i.e., the pixel is part of the background), then C_j is considered normally distributed, with mean and variance estimated individually for each background pixel (see below). If $Q_j = 1$ (i.e., the pixel is part of the foreground), then C_j is considered uniformly distributed in the color space (the least possible informative prior for the foreground is assumed). Then, the function for the factor node in level 5 is,

$$P\left(C_j \middle| Q_j\right) \sim \begin{cases} N\left(\mu_j, \sigma_j^2\right) & if\ Q_j = 0 \\ U\left([0,1]^3\right) & if\ Q_j = 1 \end{cases}.$$

The parameters $\{\mu_j\}$ and $\{\sigma_j^2\}$ of the background distributions are computed using the pixel-wise standard estimators of the mean and variance of a normal distribution, respectively, from a video of the empty scene. More sophisticated background models could be used for this part (e.g., [30]), and further improvements are expected.

In this formulation, unlike most "shape from silhouette" formulations, it is not necessary to make a hard decision about the state of the pixel (background/foreground) beforehand; the uncertainty on the pixel occupancy is transmitted to the level 3 where it is integrated with the *a priori* knowledge of the object (through the 3D prior), to make a voxel assignment decision.

3.5 Putting the Pieces Together

We have now described the relationships between the variables in a *single* ray: the color of the pixel in the ray, its occupancy, and the voxels in its line of sight. These constitute a small subset of all the variables in the system, and are represented in the (sub) graph of Figure 2. For each image pixel whose corresponding ray intersects the object's bounding box, there is a subgraph with the same structure as this one. The union of all these subgraphs constitutes the *general factor graph* where the inference is carried out (more on this in Section 4.1).

Implicit in the construction of this graph are the box registration parameters (position χ, size ρ, and orientation θ) introduced in Section 3.1. For each joint assignment of these registration parameters, a different correspondence between pixels and voxels is established, leading to a different general graph. Depending on the application, these parameters could be known (e.g., could be a user input) or unknown; in Section 5.4 we present both cases.

By simple inspection of the relationships expressed in the graph in Figure 2, it follows that the joint distribution of the variables is,

$$P_{\chi,\rho,\theta}(V_1, \ldots, V_M, Q_1, \ldots, Q_N, C_1, \ldots, C_N) = \prod_{j=1}^{N} P\left(C_j \middle| Q_j\right) \cdot \prod_{j=1}^{N} P\left(Q_j \middle| Pa\left(Q_j\right)\right) \cdot \prod_{i=1}^{M} P(V_i), \quad (5)$$

where $Pa\left(Q_j\right)$ is the set of parents of Q_j (i.e., the set of voxels in its line of sight), and the registration parameters were included as subscripts as a reminder that this probability (and the general factor graph) is computed for a particular set of values of these parameters. Our goal now can be restated as to find the 3D reconstruction (i.e., $V_1, \ldots, V_M$) and (incidentally) the reprojection (i.e., $Q_1, \ldots, Q_N$) that maximizes the conditional probability,

$$P_{\chi,\rho,\theta}(V_1, \ldots, V_M, Q_1, \ldots, Q_N | C_1, \ldots, C_N), \quad (6)$$

given a possible value for the registration parameters. The registration parameters can be estimated as the set of values (χ, ρ, θ) inside the registration parameter space, that maximizes (6). Below we describe a procedure to efficiently maximize (6).

4 Inference

The algorithms that can be used to make inferences in the general factor graph defined in Section 3.5, together with their computational complexity and the optimality guarantees of the computed solutions, strongly depend on the graph topology, in particular, on the existence of loops in this graph. This is what we investigate next. In subsequent sections, one of these inference algorithms is described in detail and then, having presented the proposed framework, it is compared with other work.

4.1 Graph Topology

As mentioned in Section 3.5, the general graph of the proposed system is the union of the subgraphs corresponding to the individual rays (through the image pixels) intersecting the bounding box. In an ideal situation, where no more than one ray intersects a voxel (Figure 4a), the general graph consist of a disjoint union of trees (a forest). In this case, exact and efficient inference can be carried out by the Sum-Product or Max-Sum algorithms [25], two instances of *belief propagation*.

As conditions start to deviate from this ideal situation, links between subgraphs appear (Figure 4b). Nevertheless, belief propagation can still be applied to find exact

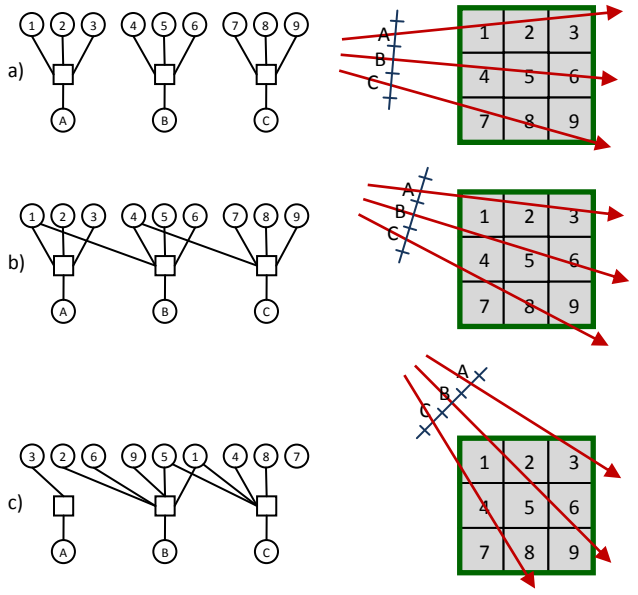


Figure 4: Topology of the graph. Only levels 2-4 are shown in the graphs on the left. (a) In an ideal situation, different rays from a single camera do not intersect the same voxels (right), and the general graph is the union of disconnected subgraphs (left), each subgraph containing a pixel and its corresponding voxels. (b) The subgraphs are now connected but the general graph still does not contain cycles. (c) The general graph contains cycles now.

solutions as long as there are no loops (note that the graph on the left of Figure 4b is a tree rooted at node 4). When many voxels are intersected by more than one ray (Figure 4c), loops in the graph generally start to appear, and weaker optimality guarantees apply [31].

The graph topology is then closely related to the number of rays intersecting the voxels, which in turn is directly related to the resolution of the input image (recall that, by definition, there is one ray per pixel). Halving the resolution of the input image (i.e., downsampling it by two in each direction) reduces the number of rays per voxel, on average, by four. Lower resolutions provide less rays per voxel, implying both less loops in the graph (desirable), and (apparently) less "information" about the state of the voxel (undesirable). Thus, which input image resolution should be used?

When downsampling the input image (see below how), information in the higher resolution rays is not discarded, rather it is integrated to yield the lower resolution rays. In other words, information from the multiple high resolution rays that intersect a voxel is *summarized* in one or a few low resolution rays. Information about the voxels is not lost, while extra loops that would appear at the higher resolution graph are avoided. Therefore, we downsample the input image until the average number of rays per voxel is in the interval [1,4). Further downsampling should be avoided, since it would result in voxels that are not intersected by any ray, and so no information is collected from them. In Section 5.3, empirical evidence supporting the prediction that this resolution is optimal is presented.

To downsample an image by two, each pixel at the new resolution is computed as the average of the corresponding 2x2 block of pixels at the original resolution. To compute the background probability from the downsampled image, the mean colors ($\{\mu_j\}$) and the variances ($\{\sigma_j^2\}$) at the new resolution are required. The mean colors are computed by downsampling the mean colors at the original resolution using the procedure just described, while the new variances are computed, assuming pixel independence, as the sum of the variances of the four original pixels that compose the new pixel.

It is always possible to obtain the situation of Figure 4a (a loopless graph), and therefore perform *exact* calculations on the graph, by defining the bounding box in such a way that two of its faces are parallel to the rays and the remaining one is perpendicular (it ceases to be a right prism, though), and then recomputing the 3D prior inside this box. Here again, the image formation process proposed in (4) is essential to translate the prior from the original to the modified box, taking into account their different geometries. We opted not to pursue this approach and study the more general case, which is important since in the case of higher order priors, the tree structure is surely lost, and the generality of the study here presented is needed.

An alternative to belief propagation, the junction tree algorithm, [32], can always be used for exact inference, even in graphs containing loops, which is the case of the graphs we encounter in general. In essence, it performs belief propagation on a modified graph whose cycles have been eliminated. The first step in the construction of this modified graph is moralization, or marring the parents of all the nodes. In the graphs we encounter, this process would create large cliques, because all the voxels in a pixel's line of sight are parents of the pixel. Since the complexity of the junction tree algorithm grows exponentially with the size of the largest clique in the modified graph, the large click size prevents the (efficient) application of this algorithm in our case. An alternative that is not guaranteed to produce exact solutions, but has provided excellent results in many cases [33], is loopy belief propagation. Loopy max-sum, an instance of loopy belief propagation, is the algorithm that we use (and optimize) for inference.

In a tree graph, like the one portrayed in Figure 4b, the following finite sequence of messages between levels guarantees that the exact global solution will be found: 1→2; 2→3; 6→5; 5→4; 4→3; and 3→2. Loops in the general proposed graph appear between levels 2 and 3, and hence, if there are loops in the graph, messages between these levels (2→3 and 3→2) have to be passed until convergence. Red arrows in Figure 2 show the direction of the flow of information in the graph: information from the actual image observations flows upwards while prior information flows downwards; they meet in the loops between levels 2 and 3 where the most time consuming part of the inference is carried out. How to optimize this inference is the subject of the next section.

4.2 Inference for 3D Reconstruction

We now show how to efficiently compute the messages that are passed as part of the max-sum algorithm. The

max-sum algorithm, [25], is a message passing process used to find the joint values of the variables, in this case $V_1, \ldots, V_M$, that maximize the conditional probability in (6). This is a *hard reconstruction* approach, in the sense that a binary decision of the state of each voxel is made. In addition, the maximum value attained by the probability is computed as part of the reconstruction, indicating how well the prior adjusts to the current observations (which is important, e.g., for object detection).

The most time consuming step of the loopy max-sum algorithm, when applied to our general graph, is to compute the messages that the projection factor nodes send to the voxel variable nodes (3→2). The message to send along one of these links (say link 1 in Figure 5, the others are computed analogously), is computed according to the formula [Equation (8.93) in [25]] (variables are defined in Figure 5):

$$\mu_{f\to v_1}(v_1) = \max_{q,v_2,\ldots,v_n} \left\{ \ln P(q|v_1,\ldots,v_n) + \mu_{q\to f}(q) + \sum_{i=2}^{n} \mu_{v_i\to f}(v_i) \right\}. \tag{7}$$

A general implementation of (7) finds the maximum by exhaustively computing the value of the expression in brackets for every possible state of the remaining n variables. Since there are n binary variables, this process has complexity $O(2^n)$. In contrast, we show next that, by exploiting the particular form of (4), this expression can be computed in polynomial time. Taking the maximum in q and renaming terms, (7) yields $\mu_{f\to v_1}(v_1) = \max[A_0(v_1), A_1(v_1)]$, where

$$A_0(v_1) \triangleq \mu_{q\to f}(0) + \max_{v_2,\ldots,v_n} \left\{ \ln P(q=0|v_1,\ldots,v_n) + \sum_{i=2}^{n} \mu_{v_i\to f}(v_i) \right\}, \tag{8}$$

$$A_1(v_1) \triangleq \mu_{q\to f}(1) + \max_{v_2,\ldots,v_n} \left\{ \ln P(q=1|v_1,\ldots,v_n) + \sum_{i=2}^{n} \mu_{v_i\to f}(v_i) \right\}. \tag{9}$$

Substituting (4) into (8), and using the distributive property of the *max* operation with respect to the *sum* operation, yields a simple $O(n)$ expression to compute $A_0(v_1)$:

$$A_0(v_1) = \mu_{q\to f}(0) - \alpha_1(v_1).r_1 + \sum_{i=2}^{n} \max_{v_i} \left[\mu_{v_i\to f}(v_i) - \alpha_i(v_i).r_i\right].$$

Before deriving an expression for $A_1(v_1)$, note that if $A_0(v_1) \geq \mu_{q\to f}(1)$, then $A_1(v_1)$ does not even need to be computed; the term in brackets in (9) is always negative and therefore $A_0(v_1) > A_1(v_1)$. This is the case for every pixel that has a high probability of being background

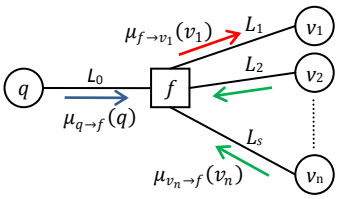


Figure 5: The most time consuming part of the belief propagation algorithm in our case: to send a message, $\mu_{f\to v_i}(v_i)$ in red, from a projection factor node f (in level 3 of Figure 2) to a voxel variable node v_i (in level 2). This factor node has already received messages from the pixel occupancy q (in level 4), $\mu_{q\to f}(q)$ in blue, and from the voxels, $\mu_{v_j\to f}(v_j)$ in green. A general implementation computes this message in $O(2^n)$ complexity (n is the number of voxels in the line of sight of the pixel), whereas our implementation has complexity $O(n^d)$.

(remember that if $P(q=1)\to 0$ then $\mu_{q\to f}(1)\to -\infty$), which shows that this algorithm gracefully reduces to a voxel carving type-algorithm (with the same complexity) when processing pixels that, with high confidence, belong to the background.

Messages corresponding to pixels that belong to the foreground, or that belong to the background with low confidence, require the computation of $A_1(v_1)$ and demand additional work. To compute $A_1(v_1)$, we first derive a more convenient expression using the following definitions:

$$\Delta_\alpha \triangleq \alpha_1 - \alpha_0 \qquad \Delta_i \triangleq \mu_{v_i\to f}(1) - \mu_{v_i\to f}(0) \qquad R \triangleq \sum_{i=1}^{n} r_{ji}$$

Substituting (4) into (9) and using these definitions, (9) can be rewritten as

$$A_1(v_1) = \mu_{q\to f}(1) + \sum_{i=2}^{n} \mu_{v_i\to f}(0) + \max_{v_2,\ldots,v_n} \left\{ \ln\left[1 - e^{-\alpha_0.R}.\exp\left(-\Delta_\alpha \sum_{i=1}^{n} v_i.r_i\right)\right] + \sum_{i=2}^{n} \Delta_i.v_i \right\}. \tag{10}$$

Let us now define the function

$$G_Z(W,V) \triangleq W + \max_{\{v_i:i\in Z\}} \left\{ \ln\left[1 - V.\exp\left(-\Delta_\alpha \sum_{i\in Z} v_i.r_i\right)\right] + \sum_{i\in Z} \Delta_i.v_i \right\}, \tag{11}$$

where Z is a set containing the indices of the voxels whose state was not yet estimated (initially all of them).

Using this definition, Equation (10) can be rewritten as,

$$A_1(v_1) = \mu_{q\to f}(1) + \sum_{i=2}^{n} \mu_{v_i\to f}(0) + G_{\{2,\ldots,n\}}(0, e^{-\alpha_0.R - \Delta_\alpha.v_1.r_1}) \tag{12}$$

An efficient recursive algorithm to compute $G_Z(W,V)$ *exactly* is described in Section SM1 of the supplemental material. The complexity of this algorithm is $O(n^d)$ where n is the number of voxels in a ray, and d is the maximum depth explored in the recursion. This depth is not uniform for all the messages, and depends on the particular class of the prior, the value of the parameters ε_0 and ε_1, and the lengths r_{ji} of the rays inside the voxels. In the experiments presented in Section 5.4 on the class of "walking people," depths of 1 and 2 were usually observed, in the fraction of cases where A_1 had to be computed at all. This quasi-linear complexity contrasts with the $O(2^n)$ complexity of the general standard implementation.

For completeness, in Section SM2 of the supplemental material we show how to efficiently run the sum-product algorithm, [25], on the general graph that was derived in previous sections, computing each message in $O(1)$. This algorithm computes a soft reconstruction, in the sense that the *a posteriori* occupancies, $\{P(V_i|\{C_j\})\}_{i=1}^{M}$, of the voxels are obtained, rather than a binary decision on their state. Soft reconstruction approaches have been previously pursued in other works (e.g., [20], [21]).

This concludes the description of the computationally efficient inference for the proposed graphical model. Having presented the framework, we now compare it with prior works.

4.3 Relation to Prior Work

Snow *et al.* [19] proposed an energy function that is minimized by solutions to the silhouette intersection problem. Expressed in our notation, this energy function is,

$$E(V_1,\ldots,V_M) = \sum_{i=1}^{M} D(V_i) + \lambda R(V_1,\ldots,V_M),$$

where $R(\cdot)$ is a smoothness (regularization) term, and $D(V_i)$ is a function that penalizes occupied voxels outside any silhouette and empty voxels inside all the silhouettes (this is a standard MRF type of energy). This formulation does not consider dependencies between voxels in the same ray, ignoring the fact that voxels in a ray "compete" to explain the outcome of its common ray (clearing a voxel in a *foreground* ray *should* increase the likelihood that other voxels in the ray are set; "someone has to take responsibility"). While it can be afforded to ignore this dependency in a multi-view setup, doing so in a single-view setup leads to very poor results.

In our formulation ((4) and levels 2 and 3), the rules of conditional independence in Bayesian Nets ([25], Section 8.2) dictate (as expected) that voxels in a common ray become conditionally dependent once the pixel in the ray (i.e., a common descendant of the voxels) is observed. This is due to the so called "explaining away" phenomenon. This dependency leads to a $O(2^n)$ complexity in the computation of the messages from the projection nodes to the voxels (3→2). Others have noted similar difficulties in related formulations (e.g., [20], [21]), but to the best of our knowledge, we are the first to provide an exact, yet computationally efficient, solution to this problem (in Section 4.2 of this article and sections SM1 and SM2 of the supplemental material).

This concludes the presentation of our framework. We now proceed to present some experimental results validating this framework.

5 Results

A dataset containing solids corresponding to 12 people walking for hundreds of frames was downloaded from [27]. This dataset also contains the original images, from five different points of view, used in the 3D reconstruction of the solids. For our experiments, the dataset is split in two: a *training dataset* containing the solids and images of 11 of the 12 people, and a *testing dataset* containing the solids and images of the remaining person. The training dataset is used to learn the 3D priors and the parameters ε_0 and ε_1, in sections 5.1 and 5.2, while the testing dataset is used to select the optimal resolution to use and evaluate the approach, in sections 5.3 and 5.4 respectively.

5.1 Learning 3D Priors for the Class of "Walking People"

Thirty-three sequences in the training dataset, each one containing one person (out of eleven) walking for hundreds of frames each, were processed to obtain a 3D prior for the class "walking people," as described in Section 3.2 (Figure 6a). The size of all the 3D priors learned and used in this work is 10x10x20 voxels (20 in the vertical direction).

To contemplate dependencies between voxels, as explained in Section 3.1, the class of "walking people" was divided into four subclasses and different 3D priors were obtained for each subclass. Each subclass corresponds to a different *pose*, or phase of the walking cycle, namely: "right leg in front," "left leg up," "left leg in front," and "right leg up". Figure 6b shows the 3D prior obtained for the class "right leg in front." Additional details on the construction of these 3D priors can be found in Section SM3 (in the supplemental material). Videos of the classes "walking people" and "walking people with right leg in front" were included as supplemental material as well. A new "registration" parameter φ, in addition to the usual χ, ρ, and θ, is introduced to account for the pose. In the following experiments the 3D priors computed for the four subclasses were used.

5.2 Parameter Estimation

In order to analyze the quality of the results, two measures are defined for the error between two solids, generally the ground truth solid, R_{GT}, and the reconstruction obtained with our proposed framework, R_i. Since solids R_{GT} and R_i are dense sets of 3D points, the usual set operations (e.g., union '∪,' intersection '∩,' and set difference '−') are defined on them, as well as the volume measure (denoted by $|\cdot|$). It is desirable that the reconstruction error, E_R, penalizes both the omitted and added volume in the reconstruction, and also that this error is expressed relative to the original volume. Then, the reconstruction error is defined as

Figure 6: Snapshots generated every 60 degrees from the 3D priors of the classes (one per row): (a) "Walking person" and (b) "Walking person with right leg in front."

$$E_R(R_i) \triangleq 100.\frac{|R_{GT} - R_i| + |R_i - R_{GT}|}{2.|R_{GT}|} = 100.\frac{|R_{GT}| + |R_i| - 2.|R_{GT} \cap R_i|}{2.|R_{GT}|}. \quad (13)$$

There are several points to notice about this definition. Firstly, the lowest possible error is zero and it is attained only when the reconstructed and ground truth volumes are identical. Secondly, this error is not bounded above and increases linearly with the added volume, assigning unboundedly large values to unboundedly large reconstructions. Thirdly, when the reconstructed and ground truth volumes are approximately equal (which is commonly the case), the error reduces to,

$$E_R(R_i) \approx 100.\left(1 - \frac{|R_{GT} \cap R_i|}{|R_{GT}|}\right),$$

which is the percentage of the original volume that is missing in the reconstruction (equal to the excess volume in the reconstruction); under this assumption the error takes values in the interval [0,100]%.

The second quality measure, introduced for completeness, is the percentage increment in reconstructed volume (compared with the original volume),

$$V_R(R_i) \triangleq 100.\left(\frac{|R_i|}{|R_{GT}|} - 1\right). \quad (14)$$

Definitions (13) and (14) are used in this section to quantify the quality of the results.

In order to fix the two parameters of the framework, namely ε_0 and ε_1, the set of values that minimized different error criteria (see below) on the training dataset were considered. Each error criterion was computed in a grid spanning eight orders of magnitude for each parameter.

The first criterion tested was minimizing the mean reconstruction error, assuming that all the registration parameters for each case were known (i.e., using the true registration parameters obtained from the true solids, as in Section 5.1). Since this error was found to be fairly insensible to the choice of these parameters (a modest 2% overall difference, between 36% and 38%), this criterion was not adopted.

The second criterion tested, that was finally adopted, consisted on minimizing the mean position error (according to the L_2 norm), assuming that all the registration parameters for each case, except the position, were known.[9] The position was estimated as the one that produced a reconstruction with the highest likelihood (6), among all positions in a 1m x 1.5m grid around the true position, with 25cm of separation between rows and columns. The position error in this case, spanned more than one order of magnitude (Figure SM3 in the supplemental material). The minimum error of 3.6cm was obtained for the pair of parameters $\varepsilon_0 = 0.056$ and $\varepsilon_1 = 0.178$. These values were used in all subsequent experiments.

5.3 Selecting the Optimal Resolution

It was predicted in Section 4.1 that using an input image resolution of more than four rays per voxel (on average), would degrade the performance of the inference algorithm by adding unnecessary loops in the system's general graph. This prediction was confirmed experimentally (see Table 1 below), by computing the mean reconstruction error and the mean volume increment on the testing dataset for three different resolutions (first column of the table). All the registration parameters in each frame were assumed to be known. The remaining columns report the reconstruction error and volume increment as defined in (13) and (14) respectively, and the computation time per frame.[10]

Table 1

Resolution interval (pixels/voxel)	E_R (%)	V_R (%)	Time (seconds)
[1, 4)	37.1	-1.8	0.52
[4, 16)	36.9	-8.1	2.00
[16, 64)	38.5	-10.2	8.67

Note that the lowest resolution, having 16 times less pixels than the highest resolution, still produced a modest 1.4% improvement in the reconstruction error and an appreciably more precise estimation of the volume (by 8.4%), in a significantly shorter time. In the comparison between the lowest and medium resolutions, the former produced a better volume estimation (by 6.3%), and a slightly worse reconstruction error (by 0.2%), in one

[9] Trying exhaustively all orientations, sizes and poses for each set of parameters would have been much more time consuming and is not expected to produce better results.

[10] The computation time measured the time that it took to construct the graph and run 5 iterations of the max-sum algorithm for one frame (image), on a single core of a 1.8 GHz AMD Turion-64 processor. These procedures were implemented in C++.

fourth of the time. Considering these results, the lowest resolution was chosen and used in the experiments described in the next section.

5.4 Reconstruction Experiments

Using the resolution and parameters selected in previous sections, solids were reconstructed for 21 frames in the testing dataset. The solids that produced the 3 lowest (frames 590, 610 and 635) and highest (frames 600, 620 and 640) reconstruction errors are shown in Figure 7. The first column in this figure contains the images, the only input to the reconstruction algorithm. The second column contains images from a different point of view taken synchronously with the images in the first column. These are included only for the purpose of evaluating the reconstruction and were not used by the algorithm. The third column are the pixel occupancies that the color model assigns to the pixels in the input image, that is, the probabilities "flowing up" in the general graph. The fourth column contains the pixel occupancies computed by projecting the reconstructed solid, these probabilities "flow down" in the graph. The fifth and sixth columns contain two views of the reconstructed solids from points of view close to the points of view in columns 1 and 2, respectively.

Observe the correctness of the reconstructions, even for the frame with the highest reconstruction error (frame 620 with E_R=48%). It is interesting to understand the source of the reconstruction error in this case. Observe in the frame that the person is considerably tilted to the left (second column), in a direction that hides this fact from the camera used for the reconstruction (first column). The magnitude of the tilt, essential for an accurate reconstruction, is very hard to judge (even for humans) from the single camera used in the reconstruction. Videos of three of these reconstructions are included as supplemental material.

Next we study the registration errors (Table 2). Experiments were performed assuming different degrees of uncertainty (first column of the table), from no unknown registration parameters (first row with results), to three unknown registration parameters (last row). A set of possible values, to use when a parameter is unknown, was defined for each parameter, as follows:

$$\chi \in \{-50\text{cm}, -25\text{cm}, \dots, 50\text{cm}\} \times \{-75\text{cm}, -50\text{cm}, \dots, 75\text{cm}\}$$
$$\theta \in \{0°, 22.5°, \dots, 337.5°\}$$
$$\rho \in \{1.65\text{m}, 1.70\text{m}, \dots, 1.85\text{m}\}$$
$$\varphi \in \{1,2,3,4\}$$

The reconstruction (as described in Section 4.2) was performed for each point in a grid defined by the Cartesian product of the possible values of the unknown parameters. For each frame, the estimate of the unknown registration parameters corresponded to the point in the grid that returned the highest likelihood, in (6). The errors in the estimation were measured using the L_2 norm for χ, θ and ρ and the L_0 norm for φ.

The results of the experiments consisted of the average, across 21 frames in the testing dataset, of the following errors: reconstruction error (2nd column), volume increment (3rd column), position error (4th column), pose error (5th column), orientation error (6th column), and size error (7th column).

Table 2

Unknown registration parameters	E_R (%)	V_R (%)	E_χ (cm)	E_φ (%)	E_θ (°)	E_ρ (cm)
-	37.1	-1.8	-	-	-	-
χ	37.2	-2.2	6.2	-	-	-
φ	37.8	-2.2	-	57	-	-
θ	37.5	-2.3	-	-	45	-
ρ	36.4	-7.4	-	-	-	5.0
χ, φ	37.7	-2.9	8.5	48	-	-
χ, φ, θ	38.1	-3.6	12.4	81	65	-

In general and as expected, the volume increase (V_R) and reconstruction error (E_R) slightly increase (in absolute value) with the uncertainty. The only exception being the case where only the size (ρ) is unknown, that produced the best reconstruction error (even better than the no-uncertainty case) at the expense of a higher error in the volume estimate.

The position and orientation errors (E_χ and E_θ) also increased with the uncertainty. When the position was the only unknown parameter, the position error was half of that when two other parameters (φ and θ) were unknown. In the case of the orientation error, adding two other unknown parameters (χ and φ) only increased the error by 44%.

The pose error is comparatively the highest, in one case being even worse than pure chance. Part of this error is attributable to the fact that some poses are indistinguishable from the silhouette alone (there is not enough information to determine such poses). Integrating information from a few consecutive frames may mitigate this problem.

6 Conclusions and Future Work

In this work we introduced a novel entity, the 3D prior, to encode the distribution of mass in 3D space of the objects of a given class. An interpretation equating the 3D priors with semitransparent objects was subsequently established, and a law to project these entities to the 2D image plane was postulated by analogy to the Beer-Lambert law in optics. This framework readily translates into a graphical model that naturally integrates prior information with actual observations from a single input image. It was demonstrated that efficient inference in this graphical model is possible, by presenting a novel algorithm to achieve it. It was also demonstrated that optimal solutions can be guaranteed under certain conditions, which were enunciated. Experimental results demonstrated the accuracy of the approach for 3D reconstruction, localization and volume estimation.

The proposed framework can be extended in a number of ways, some of which were already stated above. Higher order 3D priors (sections 3.1 and 3.2) can be used to

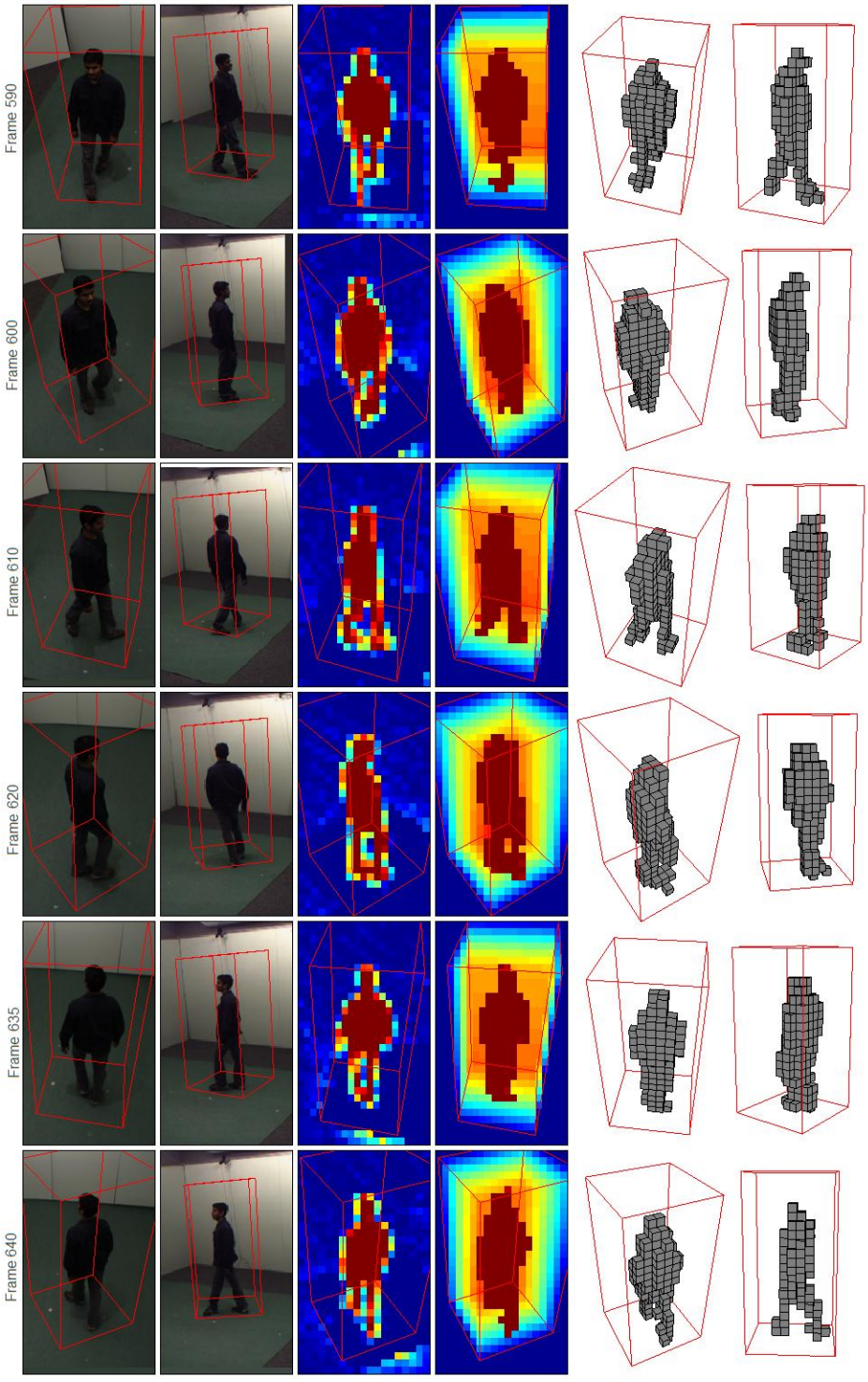


Figure 7: Reconstructions for six frames in the testing dataset when the registration parameters are known. The best and worst three reconstructions (according to reconstruction error) are shown in the first and last three rows, respectively. The bounding boxes were included in all images (in red).

1st column: single input image used in the reconstruction.

2nd column: an orthogonal view, not used in the reconstruction, included only for comparison.

3rd column: pixel occupancies computed from the single input image using the color model (blue is 0, dark red is 1).

4th column: pixel occupancy probabilities predicted by the 3D prior and estimated reconstruction.

5th and 6th columns: two views of the reconstructed solid, corresponding approximately to the views in columns 1 and 2.

improve the reconstruction from a single image. Temporal 3D priors (Section 3.2), by modeling the dynamics of the distribution of mass in space, could integrate information from multiple (consecutive) frames, further improving upon single frame reconstruction. Multiscale 3D priors (Section 3.3), on the other hand, may further accelerate the inference by matching the required level of detail with some measure of the reconstruction error.

Other alternatives to extend this work, which have not already been mentioned, include: 1) Handling multiple views, or multiple projections of the object that appear in the same view (e.g., shadows or reflections). In this case, however, loops in the general graph surely appear, and the optimality of the solutions found cannot be guaranteed. Whether the inference framework proposed will actually find good solutions is an open question. 2) Exploiting the information "inside the silhouette" that is currently discarded, for example, by incorporating in the 3D prior information about the colors that are expected in certain voxels (e.g., voxels corresponding to the face or hands), or information indicating that some voxels are more likely to have similar colors (e.g., the colors in the

voxels of one leg are probably similar to the colors in the voxels of the other leg).

An interesting connection exists between the present work and some recent findings in neuroscience. Bryant [34] pointed out that many hippocampal and cortical neurons have been observed to possess *place fields*; these cells respond to fixed positions in egocentric (i.e., relative to the *self*'s coordinate system) or allocentric (i.e., relative to the *environment*'s coordinate system) space rather than specific objects in those positions, indicating the existence of specialized *place coding* neurons. These neurons could be playing the role of the voxels in our framework, providing the basis for a neural-inspired implementation of the algorithms we described. This connection deserves further investigation.

ACKNOWLEDGMENT

Support for this work came from NSF, ONR, NGA, ARO, and DARPA.

REFERENCES

[1] Rother, D., Patwardhan, K., Aganj, I. and Sapiro, G., "3D Priors for Scene Learning from a Single View." *S3D Workshop (at Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition)*. 2008.

[2] Dyer, C. R., "Volumetric scene reconstruction from multiple views." Book: L. S. Davis. *Foundations of Image Understanding*. Kluwer, Boston. 2001, pp. 469-489.

[3] Slabaugh, G., Culbertson, B., Malzbender, T. and Schafer, R., "A Survey of Methods for Volumetric Scene Reconstruction from Photographs." *Proceedings of the Joint IEEE Tcvg and Eurographics Workshop in Volume Graphics*. 2001.

[4] Lazebnik, S., Furukawa, Y. and Ponce, J., "Projective Visual Hulls." *Int'l J. Computer Vision*. 2007.

[5] Aganj, E., Pons, J.P., Ségonne, F. and Keriven, R., "Spatio-Temporal Shape from Silhouette using Four-Dimensional Delaunay Meshing." *Proc. IEEE Int'l Conf. Computer Vision*. 2007.

[6] Han, F. and Zhu, S. C., "Bayesian Reconstruction of 3D Shapes and Scenes From A Single Image." *Proceedings of the First IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*. 2003.

[7] Ramanan, D., "Learning to Parse Images of Articulated Objects." *Advances in Neural Information Processing Systems*. 2006.

[8] Bowden, R., Mitchell, T. A. and Sarhadi, M., "Reconstructing 3D Pose and Motion from a Single Camera View." *Proceedings of the British Machine Vision Conference*. 1998.

[9] Howe, N. R., Leventon, M. E. and Freeman, W. T., "Bayesian Reconstruction of 3D Human Motion from Single-Camera Video." *Advances in Neural Information Processing Systems*. 2000.

[10] Sigal, L. and Black, M. J., "Predicting 3D people from 2D pictures." *IV Conference on Articulated Motion and Deformable Objects*. 2006.

[11] Scharstein, D. and Szeliski, R., "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms." *Int'l J. Computer Vision*. 2002.

[12] Wang, J. J. L. and Singh, S., "Video analysis of human dynamics - a survey." *Real Time Imaging*. 2003.

[13] Mikic, I., Trivedi, M., Hunter, E. and Cosman, P., "Human Body Model Acquisition and Tracking Using Voxel Data." *Int'l J. Computer Vision*. 2003.

[14] Isard, M. and MacCormick, J., "BraMBLe: A Bayesian Multiple-Blob Tracker." *Proc. IEEE Int'l Conf. Computer Vision*. 2001.

[15] Agarwal, A. and Triggs, B., "Learning to Track 3D Human Motion from Silhouettes." *Proc. of the 21st Int'l Conf. on Machine Learning*. 2004.

[16] Seemann, E., Leibe, B. and Schiele, B., "Multi-aspect detection of articulated objects." *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*. 2006.

[17] Balan, A.O., Sigal, L., Black, M.J., Davis, J.E. and Haussecker, H.W., "Detailed Human Shape and Pose from Images." *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*. 2007.

[18] Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J. and Davis, J., "SCAPE: Shape completion and animation of people." *Proc. of SIGGRAPH*. 2005.

[19] Snow, D., Viola, P. and Zabih, R., "Exact Voxel Occupancy with Graph Cuts." *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*. 2000.

[20] Franco, J. S. and Boyer, E., "Fusion of Multi-View Silhouette Cues Using a Space Occupancy Grid." *Proc. IEEE Int'l Conf. Computer Vision*. 2005.

[21] Broadhurst, A., Drummond, T. W. and Cipolla, R., "A Probabilistic Framework for Space Carving." *Proc. IEEE Int'l Conf. Computer Vision*. 2001.

[22] Baumberg, A. M. and Hogg, D. C., "An efficient method for contour tracking using active shape models." *Proc. of the Workshop on Motion of Nonrigid and Articulated Objects*. 1994.

[23] Savarese, S. and Li, F., "3D generic object categorization, localization and pose estimation." *Proc. IEEE Int'l Conf. Computer Vision*. 2007.

[24] Kushal, A., Schmid, C. and Ponce, J., "Flexible Object Models for Category-Level 3D Object Recognition." *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*. 2007.

[25] Bishop, C. M., *Pattern Recognition and Machine Learning*. Springer, 2006.

[26] Wang, J. and Cohen, M., "Image and Video Matting: A Survey." *Foundations and Trends in Computer Graphics and Vision*. 2007.

[27] Inria., INRIA Xmas Motion Acquisition Sequences (IXMAS). *The Multiple-Camera/Multiple-Video Database of the PERCEPTION group. 2006*. Online: https://charibdis.inrialpes.fr/html/non-secure-sequence.php?s=IXMAS.

[28] Lakowicz, J. R., *Principles of Fluorescence Spectroscopy*. New York : Springer Science+Business Media, LLC, 2006.

[29] Szeliski, R., "Rapid octree construction from image sequences." *CVGIP: Image Understanding*. 1993.

[30] Mittal, A. and Paragios, N., "Motion-based background subtraction using adaptive kernel density estimation." *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*. 2004.

[31] Weiss, Y. and Freeman, W.T., "On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs." *IEEE Trans. on Information Theory*. 2001.

[32] Lauritzen, S. L. and Spiegelhalter, D. J., "Local computations with probabilities on graphical structures and their application to expert systems." *J. of the Royal Statistical Society*. B, 1988.

[33] Frey, B and MacKay, D., "A Revolution: Belief Propagation in Graphs With Cycles." *Advances in Neural Information Processing Systems*. 1998.

[34] Bryant, D. J., "Representing Space in Language and Perception." *Mind & Language*. 1997.

This document contains supplemental information in order to provide additional details, theoretical results, and experimental videos. The following videos were included as supplemental material:

PriorWalkingPeople.avi	- Video showing the 3D prior for the class "walking people."
PriorRightLegFront.avi	- Video showing the 3D prior for the class "walking people with their right leg in front."
Reconstruction590.avi	- Video of the 3D Reconstruction of frame 590.
Reconstruction600.avi	- Video of the 3D Reconstruction of frame 600.
Reconstruction640.avi	- Video of the 3D Reconstruction of frame 640.

SM1. Computation of $G_Z(W,V)$

This section proposes an efficient algorithm to compute $G_Z(W,V)$. This is obtained exploiting the particular form of (11) in two ways: 1) using bounds to directly (i.e., without exhaustively computing all the possible values of the variables) determine some of the variable states, and 2) defining a recursive expression for (11) that can be efficiently computed to determine the state of the remaining variables.

The state of some variables in (11) can be directly computed, reducing the dimensionality of the state space that has to be searched. To see this, let us first define the increment to the term inside the max in (11) produced by switching v_k from 0 to 1, for a particular state $\boldsymbol{v}_{Z-k}$ of the remaining variables,

$$U_Z^k(\boldsymbol{v}_{Z-k}) \triangleq \ln\left[1 - V.\exp\left(-\Delta_\alpha \sum_{\substack{i\in Z,\\ i\neq k}} v_i.r_i\right).\mathrm{e}^{-\Delta_\alpha.r_k}\right] - \ln\left[1 - V.\exp\left(-\Delta_\alpha \sum_{\substack{i\in Z,\\ i\neq k}} v_i.r_i\right)\right] + \Delta_k. \qquad \text{(SM1)}$$

Let us also define the variables Y_k^{min} and Y_k^{max} $(k \in Z)$ as the minimum and maximum possible values of (SM1), respectively, for any possible state of the remaining variables,

$$Y_k^{min} \triangleq \min_{\boldsymbol{v}_{Z-k}} U_Z^k(\boldsymbol{v}_{Z-k})$$
$$Y_k^{max} \triangleq \max_{\boldsymbol{v}_{Z-k}} U_Z^k(\boldsymbol{v}_{Z-k})$$

Then, $Y_k^{min} \geq 0$ guarantees that v_k must be set to 1, since any solution that has $v_k = 0$ can only be improved by "turning on" v_k. Analogously, $Y_k^{max} \leq 0$ guarantees that v_k must be set to 0, since any solution that has $v_k = 1$ can only be improved by "turning off" v_k. The usefulness of Y_k^{min} and Y_k^{max} stem from the fact that they can be directly computed, without computing the value of $U_Z^k(\boldsymbol{v}_{Z-k})$ for every possible state $\boldsymbol{v}_{Z-k}$ (using the fact that the function $\ln\left(\frac{1-\beta x}{1-\beta}\right)$ is strictly increasing in $\beta \in (0,1)$, for $x \in (0,1)$):

$$Y_k^{max} = \ln\left(\frac{1 - V.\mathrm{e}^{-\Delta_\alpha.r_k}}{1 - V}\right) + \Delta_k$$

$$Y_k^{min} = \ln\left(\frac{1 - V.\exp\left(-\Delta_\alpha \sum_{\substack{i\in Z,\\ i\neq k}} r_i\right).\mathrm{e}^{-\Delta_\alpha.r_k}}{1 - V.\exp\left(-\Delta_\alpha \sum_{\substack{i\in Z,\\ i\neq k}} r_i\right)}\right) + \Delta_k$$

Then, the states of some variables in Z can be defined by the following rule:

Table SM1	
If $Y_k^{min} \geq 0$	Set: $v_k := 1,\ W := W + \Delta_k,\ V := V.e^{-\Delta_\alpha.r_{jk}}$ and remove k from Z.
Else if $Y_k^{max} \leq 0$	Set $v_k := 0$ and remove k from Z.
Else	v_k is still unknown, therefore keep k in Z.

If after applying these rules to all the missing variables, there still exist some variable states that could not be estimated (i.e., some voxels fall in the third case of Table SM1), these are estimated efficiently (yet still exactly) by using the second strategy, through the recursion described next.

Note that if one of the voxels (e.g., v_1) is known to be opaque (i.e. $v_1 = 1$), then the value of the logarithm in G_Z depends only slightly on the state of the other voxels. Based on this observation, the state space to search can be split into subspaces having each one of the unknown coordinates set to one (see Figure SM1), compute the maximum in each subspace (recursively), and then combine the partial results to obtain the global result. To guarantee that the whole space is searched, the subspace that has all the (remaining) coordinates set to zero also has to be examined. The following recursion formalizes this idea ($m \triangleq |Z|$),

$$G_Z(W,V) = \max\left[G_{Z-z_1}\left(W+\Delta_{z_1}, V.e^{-\Delta_\alpha . r_{z_1}}\right), \ldots, G_{Z-z_m}\left(W+\Delta_{z_m}, V.e^{-\Delta_\alpha . r_{z_m}}\right), G_\emptyset(W,V)\right]. \quad \text{(SM2)}$$

At each level of this recursion there is one more state set to 1, compared to the previous level, making the logarithm at the current level even more insensitive to the states of the rest of the variables.

In summary the computation of $G_Z(W,V)$ has two stages: 1) the variables in Z are checked against the rules in Table SM1, and some are estimated directly and removed from Z; 2) the value of G_Z is computed recursively using (SM2) and the remaining variables in Z. These two strategies lead to the efficient computation of $G_Z(W,V)$ in $O(n^d)$ complexity, where d is the maximum depth inspected in the recursion, instead of the $O(2^n)$ complexity of the general implementation.

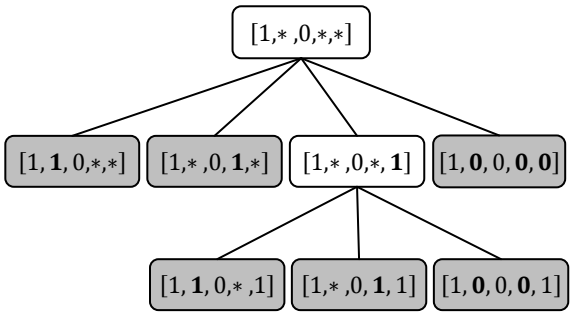


Figure SM1: An example of how the space of voxel states in a ray is searched efficiently. Each box represents all the states that match its label (the asterisks can be either 0 or 1, hence the root box contains 8 configurations). The maximum in the grayed boxes can be directly computed using the rules in Table SM1. The rest of the boxes are searched recursively. Numbers fixed last appear in bold.

SM2. Soft reconstruction

We now show how to efficiently run the sum-product algorithm on the general graph derived in sections 3.5 and 4.1. The sum-product algorithm, [SM1], is a message passing process used to marginalize the voxel variables (one at a time) from (6), summing away all other variables. The most time consuming step of the sum-product algorithm, when applied to the class of graphs we obtain, is to compute the messages that the projection factor nodes send to the voxel variable nodes (3→2). The message to send along one of these links (say link 1 in Figure 5, the others are computed analogously), is computed according to the formula [Equation (8.66) in [SM1]]:

$$\mu_{f\to v_1}(v_1) = \sum_{q,v_2,\ldots,v_n} P(q|v_1,\ldots,v_n).\mu_{q\to f}(q).\prod_{i=2}^{n}\mu_{v_i\to f}(v_i), \quad \text{(SM3)}$$

where q is a pixel occupancy variable and $v_1,\ldots,v_n$ are the voxels in its ray, f is the factor node connecting them, and $\mu_{x\to y}$ is a message to pass (or that was already passed) from x to y.

A general implementation that computes the sum in (SM3) by calculating summands for every possible state of the n variables in the summation has complexity $O(2^n)$, since there are n binary variables. In contrast we show next that, by exploiting the particular form of (4), this expression can be computed in constant time.

Opening the summation in q, and substituting (4) in (SM3), yields

$$\mu_{q\to f}(0)\sum_{v_2,\ldots,v_n}\prod_{i=1}^{n}e^{-\alpha(v_i).r_i}.\prod_{i=2}^{n}\mu_{v_i\to f}(v_i)+\mu_{q\to f}(1)\sum_{v_2,\ldots,v_n}\left[1-\prod_{i=1}^{n}e^{-\alpha(v_i).r_i}\right].\prod_{i=2}^{n}\mu_{v_i\to f}(v_i).$$

By rearranging terms and exchanging summation and product wherever possible, we obtain

$$\mu_{q\to f}(0).e^{-\alpha(v_1).r_1}.\prod_{i=2}^{n}\sum_{v_i}e^{-\alpha(v_i).r_i}.\mu_{v_i\to f}(v_i)$$
$$+\mu_{q\to f}(1)\left[\prod_{i=2}^{n}\sum_{v_i}\mu_{v_i\to f}(v_i)-e^{-\alpha(v_1).r_1}\prod_{i=2}^{n}\sum_{v_i}e^{-\alpha(v_i).r_i}.\mu_{v_i\to f}(v_i)\right]$$

Using normalized messages (such that $\sum_{v_i}\mu_{v_i\to f}(v_i)=1$), and defining the auxiliary variables

$$A_j=\prod_{\substack{i=1,\\ i\neq j}}^{n}\sum_{v_i}e^{-\alpha(v_i).r_i}.\mu_{v_i\to f}(v_i),$$

the final expression for the message is

$$\mu_{f\to v_j}(v_j)=e^{-\alpha(v_j).r_j}.A_j\left[\mu_{q\to f}(0)-\mu_{q\to f}(1)\right]+\mu_{q\to f}(1) \qquad \text{(SM4)}$$

The n messages from the factor node to the voxel variables can be computed in $O(n)$ by using (SM4), in other words, each message is computed in $O(1)$.

SM3. Details on learning the 3D priors for the class "walking people"

Each training sequence contains more than a hundred solids, acquired at consecutive time steps. In this situation, the person's walking direction can serve as the orientation parameter (θ) needed for the registration. To determine the walking direction, the trajectory (i.e., the sequence of positions) was low pass filtered and then differentiated. Figure SM2a shows the positions and orientations obtained for each frame. The remaining registration parameters

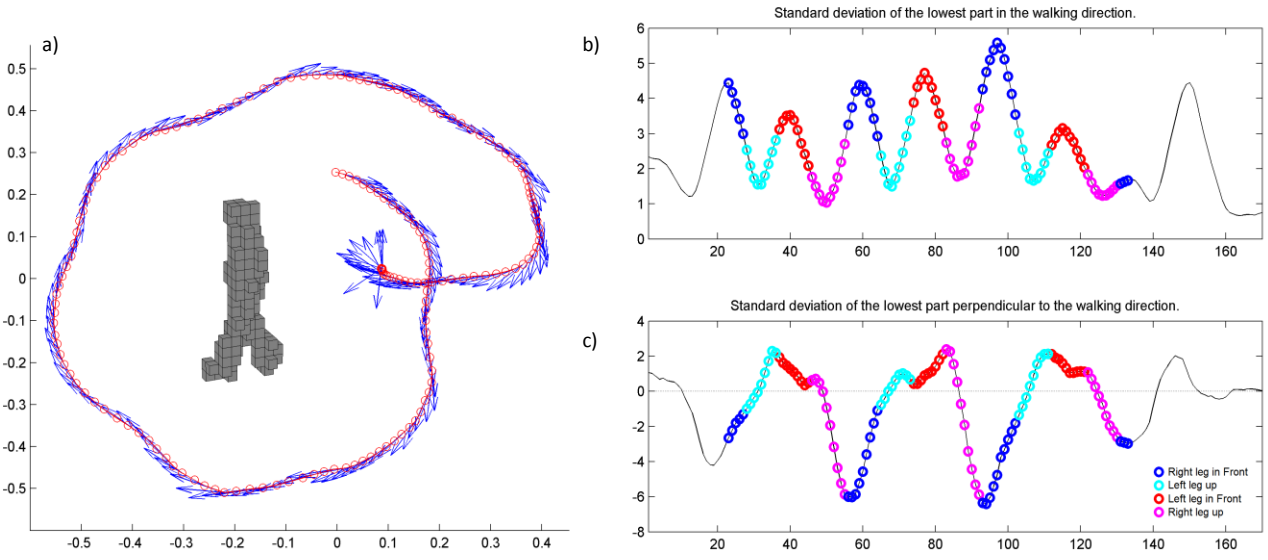


Figure SM2: Registering and classifying the 3D volumes before computing the 3D prior. a) The trajectory of the person (i.e., the sequence of its locations) is marked with red circles and the orientations are marked as blue vectors. An example of a registered and resampled solid is shown in the center. b & c) The first (b) and second (c) features used to classify the phase of the walking cycle. The colors describe the subclass assignment for each volume. See text for details.

were estimated, and the solids normalized with respect to the bounding box, as described in Section 3.2. The center of Figure SM2a shows one of these normalized solids. The resulting 3D prior was obtained by averaging these normalized solids.

Instead of averaging all the normalized solids to obtain a single 3D prior, these solids were classified into four different categories, and each category was averaged separately to obtain one 3D prior per category. The categories correspond to four different phases of the walking cycle, or poses. The normalized solids were classified automatically into one of these poses by using two features: 1) the standard deviation of the voxels in the lowest quarter of the volume, in the walking direction (see Figure SM2b); and 2) the difference in the number of occupied voxels in the lowest quarter of the solid, between the front-right and back-left and those in the front-left and back-right (see Figure SM2c).

SM4. Additional figures

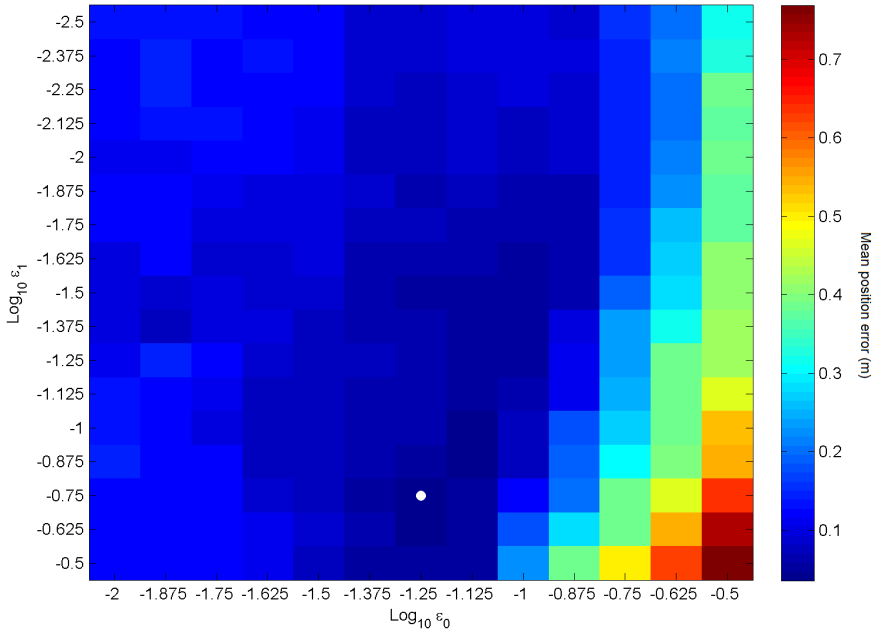


Figure SM3: Position error for different pairs $(\varepsilon_0, \varepsilon_1)$ of parameters. The pair that produced the minimum position error (marked with a white 'o') was selected.

References

[SM1] Bishop, C. M., *Pattern Recognition and Machine Learning.* Springer, 2006.