International Journal of Computing and Digital Systems
-- An International Journal

A Multi-view 3D Shape Reconstruction System using Level Sets

Moumen T. El-Melegy[1] and Nagi H. Al-Ashwal[2]

[1]Electrical Engineering Department, Assiut University, Assiut 71516, Egypt
moumen@aun.edu.eg
[2]Electrical Engineering Department, Ibb University, Ibb, Yemen
nlashwal@yahoo.com

Abstract: The 3D reconstruction of a scene from multiple images is a fundamental problem in the field of computer vision. Existing methods can be classified into two strategies: bottom-up or top-down. This paper presents a full system for complete 3D shape reconstruction following the top-down strategy. A rotary table is employed to change a camera's viewing direction to an object on the table. This offers a cost-effective solution to the multi-view stereo acquisition problem without the need for using several cameras. From the acquired calibrated images of the object, a variational approach is developed for 3D shape reconstruction of the object. The approach works directly in 3D Euclidean space based on a level set formulation. A correlation criterion between the 2D images is optimized by driving the evolution of the surface using the corresponding Euler-Lagrange equation. Several successful experiments to evaluate the proposed system are reported.

Keywords: Level set methods, Multi-view stereo and 3D reconstruction**.**

I. INTRODUCTION

The 3D reconstruction of a scene from multiple images is a fundamental problem in the field of computer vision. There are many real world applications of 3D models, such as computer graphics, robot navigation, TV/film special effects, computer games, virtual reality inspection, navigation, and object identification. Recently it has become a very important and fundamental step in particular for cultural heritage digital archiving. The motivations are different: documentation in case of loss or damage, virtual tourism and museum, education resources, interaction without risk of damage, and so forth.

Given a set of images of a scene captured from multiple calibrated cameras, the goal is to recover the unknown 3D structure using these images and the knowledge of the camera geometry. This problem is known as multi-view stereo in computer vision and is of great importance as it provides a way to infer the geometric and photometric properties of a scene without interfering with it. Additional advantages of using multi-view stereo to infer these scene properties include that the setup is fairly simple, the cost is relatively low and the process can be easily automated.

In most applications in the field of vision-based 3D reconstruction [1][4], two strategies can be applied:

- Bottom-up or data driven strategies [4][5]: These algorithms are based on image matching, using either intensity-based (direct) methods or feature-based methods. This class includes all techniques that compute correspondences across images and then recover 3D structure by triangulation and surface fitting. Establishing correspondence in this traditional way is a hard problem that often leads to many outliers [1]. Some disadvantages of this approach are that, it is computationally intensive and algorithmically demanding. Furthermore, views must often be close together (i.e., small baseline) so that correspondence techniques are effective. Correspondences must be maintained over many views spanning large changes in viewpoint. The modeling of occlusions is complicated and there is not a standardized and widely accepted framework for modeling occlusions for bottom-up methods [4][5]. There are many methods in literature belonging to this strategy. Refer to [1][4][27]for a recent review and taxonomy of algorithms.
- Top-down or model driven strategies [21]: In this case, the computations are performed in three-dimensional scene space in order to construct the volumes or surfaces in the world that are consistent with the input images. Top-down approaches assume there is a known, bounded area in which the objects of interest lie. These strategies overcome the disadvantages of the bottom-up approach; they have the ability to explicitly model occlusions and consider multiple views. Example shere include shape form silhouettes [12][13], space carving [6][15][26], reconstruction using variational methods [8][17][18][28][29], volumetric graph-cuts [30] and continuous global optimization [6].

The existing approaches suffer from one or more of the following issues in the problem of multi-view 3D reconstruction: shape representation, objective function optimization and the initialization requirements. In this paper we present a complete approach addressing these issues following the top-down strategy due to its notable advantages over the bottom-up strategies. Many existing systems [6][15][26][30] represent a shape as a set of voxels or polygon meshes, which may fail with complex shapes. In contrast, our proposed approach uses a level set representation that provides several advantages over traditional object representations, such as its capability to model complex surfaces and to cope with varying object topologies [3]. In terms of optimization, several existing approaches iteratively

update the shape based on local decisions starting from an initial volume, such as space carving methods [6][15][26], which often causes the propagation of unrecoverable errors from one region to another. In contrast, our approach defines a *global* objective function on the whole 3D space, whose optimization leads naturally to the evolution to a partial differential equation (PDE) of the level set function. This evolution is efficiently solved on a discretized 3D grid using well-defined numerical methods [20][23]. While many algorithms [12][13][8][17][18][28][29] require a good initialization to guarantee algorithm convergence, our approach tends to work starting from any initial position, and the reconstruction process is not sensitive to the initial level set function.

Our approach is inspired by Faugeras and Keriven [8], who were the first to combine image matching and shape reconstruction in a variational framework that minimizes an energy functional that is written as the integral of a data fidelity criterion on the unknown surface. However, the numerical implementation is rather complicated and requires simplification by dropping some terms. The final result tends to be sensitive to where the level set function has started its evolution. In contrast, we start with a new, different formulation that models the surface to be reconstructed as a level set embedded in an energy functional from the beginning. An advantage of this formulation is that one can easily and in straightforward manner model any available information on the object shape into the energy functional. For example, this may allow the reconstruction of an object with shape variations consistent with a set of training model examples [24]. This can prove very useful indeed in several nowadays applications of computer vision and graphics that focus on building 3D models of a certain category of objects. For example, generation of realistic 3D human face models and facial animations can indeed exploit the earlier knowledge that the object looks like a human face. Another example is the 3D modeling of the human jaw from a sequence of intra-oral images [25], which can make use of prior information on the shape of human teeth.

In addition to the distinctive and novel aspects of our approach (strong, flexible shape representation and efficient global optimization algorithm with no special initialization requirement), we build a simple, yet effective 3D reconstruction system consisting of a rotary-table and a USB camera, both controlled via a desktop PC. The system uses the rotary table to change the camera's viewing direction to an object on the table. This offers a cost-effective solution to the multi-view stereo acquisition problem without the need for using several cameras. A checkerboard calibration pattern is used to calibrate the camera in the very first view to recover the camera projective geometry and its parameters. Then the camera parameters are automatically updated for all the other views, without the need for re-calibration. The images acquired from the different views are used for shape modeling. The developed approach is successfully evaluated in several experiments using synthetic and real datasets, as well as using our own system setup.

The rest of this paper is organized as follows. In Section II we describe our acquisition setup for capturing images from multiple views. A brief description of camera geometry and calibration is also given. We explain our approach for 3D reconstruction in Section III. The experimental results are reported in Section IV. This paper is concluded in Section V.

II. SYSTEM SETUP AND CAMERA CALIBRATION

We have developed a data acquisition setup to capture *calibrated* images from multiple views of objects. The hardware setup consists of a rotary table, a USB camera and a desktop PC. The rotary table is built from scratch using a stepper motor, a driver and an interface circuit to the parallel port of the PC. Figure 1 shows the constructed rotary table. The system operates as follows: The object to be reconstructed is placed on the rotary table. In order to obtain different viewpoints of the object, we simply rotate the table by a desired angle each time and grab an image. This is repeated for a number of times (typically 10-12) to cover the object from all views. A complete program with graphical user interface (GUI), written in Visual C++, is used to control the speed, direction and the step size of the table motion and to acquire the images from the USB camera. By construction, the step size of the table rotation can be as low as 0.1°.

Most 3D reconstruction techniques require the calibration of the camera, especially if quantitative measurements are sought. Camera calibration allows us to derive the projection equations linking points in our 3D world to their projections on the image and solve for the camera intrinsic and extrinsic parameters. According to the most common camera model in computer vision, the pinhole camera model [9], a world 3D point $\tilde{\mathbf{X}} = (X,\ Y,\ Z,\ 1)^T$ and the corresponding image point $\tilde{\mathbf{x}} = (u,\ v,\ w)^T$ (the over-symbol ~ denotes homogeneous vectors) are related via

$$\tilde{\mathbf{x}} = \mathbf{P}\,\tilde{\mathbf{X}}, \tag{1}$$

where $\mathbf{P}$ is a 3×4 homogeneous camera projection matrix which can be decomposed into [9]:

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R} \mid \mathbf{t}\right], \tag{2}$$

where $\mathbf{K}$ is the camera calibration matrix containing the intrinsic parameters of the camera (e.g., focal length and principal point), $\mathbf{R}$ is an orthonormal rotation matrix and $\mathbf{t}$ is a 3D translation vector. Both $\mathbf{R}$ and $\mathbf{t}$ represent the camera extrinsic parameters which define the camera orientation and position with respect to the world coordinates. The above formulation relies on the assumption that the map from the world to the image is linear projective. That is, there exists no significant lens distortion or it has been corrected [22].

The procedure of calculating the camera matrix $\mathbf{P}$ is called camera calibration [9]. Many methods are available in literature [9] for the determination of the camera matrix. We use for this sake in our system Robert's technique [10], which has the advantage of high accuracy without the need of accurate feature extraction. This is done with the help of a checkerboard pattern, see Figure 2. The world coordinate system is chosen along the sides of the calibration pattern. At the initial viewpoint (first position of the table), the camera is initially calibrated by putting the calibration pattern on the table such that the world's Y-axis coincidences with the rotation axis of the table (A correction procedure is carried out here to compensate for any possible misalignment [2]). This way the initial projection matrix $\mathbf{P}_0$ at this first viewpoint is calibrated.

At each subsequent rotation angle, we need to compute a new projection matrix for the new viewpoint. Recalibrating the camera using the calibration pattern is not needed. By rotating the object on the table by a specific angle θ, the camera intrinsic parameters are expected not to change, but we

simply need to update the extrinsic parameters of the camera to reflect the new camera orientation.

Figure 1. The rotary table is used in our work to change the camera's viewing direction to an object.

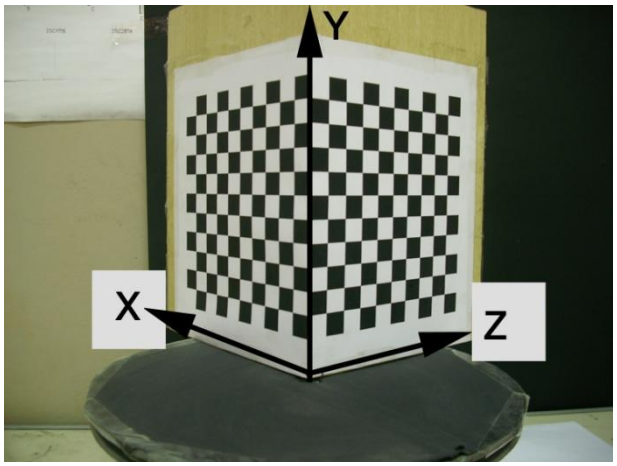

Figure 2. A checkerboard pattern is placed on the rotary table to calibrate the camera. The world's Y-axis coincidences with the table rotation axis.

The camera projection matrix at any view *i* is calculated as

$$\mathbf{P}_i = \mathbf{P}_0 \mathbf{R}(\theta_i), \tag{3}$$

where $\mathbf{R}(\theta_i)$ is a 4×4 rotation matrix around the Y-axis by the angle θ_i.

A simple experiment was carried out to assess the accuracy of camera calibration across multiple-views. Several images of the pattern from various views are used to reconstruct the corners of the pattern squares in the 3D world coordinates. These 3D points are then compared to the known ground-truth positions of the calibration pattern squares. The root-mean-square error is found to be about 0.1±0.04 *mm* across the various views, which indicates very high calibration accuracy.

III. PROPOSED APPROACH

In this paper we consider the problem of recovering the 3D shape of a given object from a set of 2D images taken from n multiple calibrated viewpoints. We assume that the object is made of Lambertian materials and there is texture in the albedo. The approach proposed in this work shares some fundamental points with that of [8]. The two approaches find a surface that minimizes an energy functional that is written as the integral of a data fidelity criterion on the unknown surface. This criterion is based on the normalized cross-correlation between image pairs. However ours is different in a number of aspects. In [8] the normalized cross-correlation between image pairs is done by picking a window around a point in one image and comparing it with its transformed window (around the corresponding point) in another image. This transformation is computed through the unknown surface, which is then taken locally planar. Here, we will present a better matching process. Instead of considering the transformations between image patches, we project a point on the surface to each visible image and then pick a window around each projected point. Then the matching is done by comparing each pair of these widows.

The Euler-Lagrange equation that minimizes the functional of [8] is driven in terms of the surface, then the evolution is implemented via introducing *later* a level set formulation with some simplification by dropping some terms. In this work, the energy functional is embedded in a level-set framework from the beginning. An advantage of this formulation is that one can easily and in straightforward manner model any available information on the object shape into the energy functional. The minimization of the energy functional is done on the whole 3D space, leading naturally to the evolution to a partial differential equation (PDE) of the level set function. All the terms in the PDE are used to implement the evolution without dropping any of them. Consequently, our approach offers a number advantages over that of [8] and similar works, e.g., [17][18][29], which tend to be sensitive to the initial position of the surface. On the contrary, our approach tends to work starting from any initial position, and the final result is not sensitive to where the level-set function starts its evolution.

In this section, we give the full details of the proposed approach. We start with brief review of some fundamental and preliminary concepts on level sets. Then we formulate our approach in a level-set framework, addressing the object visibility and evolution issues.

A. Preliminary Concepts

Level set methods were devised by Osher and Sethian [3] to *implicitly* model evolving interfaces in two or three dimensions. Level set methods have several advantages compared to the *explicit* active contours (snakes) introduced by Kass et al. [11] and other deformable methods that use the parametric representation of curves and surfaces:

- One can perform numerical computations involving curves and surfaces on a fixed Cartesian grid without having to parameterize these objects.
- It is conceptually straightforward to move from two to three and even higher dimensions (although computational cost is exponential in dimension).
- Moving interfaces automatically handle the topological changes, which happen often and are desired in evolutions; they can easily merge or separate.
- Geometric quantities are easy to calculate: surface normal, curvature, direction and distance to the nearest point on the surface. Surface motion can depend on them.

Therefore, level set methods have received attention in many fields, including image processing, computer graphics, computer vision, fluid mechanics, and computational

geometry. By now, level set methods have become standard tools for implementing evolution PDEs.

In an implicit formulation the interface in $\Omega \subset R^n$ is a non empty subset defined by [3]

$$\Gamma = \{\mathbf{x} \in \Omega; \phi(\mathbf{x}) = 0\}, \quad (4)$$

where the function, $\phi(x)$, $\phi : R^n \to R$, has the following properties

$$\begin{cases} \phi(\mathbf{x}) > 0 & for \quad \mathbf{x} \in \text{inside}(\Gamma), \\ \phi(\mathbf{x}) < 0 & for \quad \mathbf{x} \in \text{outside}(\Gamma), \\ \phi(\mathbf{x}) = 0 & for \quad \mathbf{x} \in \Gamma. \end{cases} \quad (5)$$

In three spatial dimensions the gradient of the implicit function $\phi(\mathbf{x})$, is given by $\nabla\phi = \left(\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}, \frac{\partial\phi}{\partial z}\right)$. $\nabla\phi$ is perpendicular to the isocontours of ϕ pointing in the direction of increasing ϕ. Thus, the unit normal is $\boldsymbol{n} = \frac{\nabla\phi}{|\nabla\phi|}$, for points on the interface. The mean curvature of the interface is defined as the divergence of the normal, $\kappa = \nabla.\boldsymbol{n} = \text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right)$, so that $\kappa > 0$ for convex regions, $\kappa < 0$ for concave regions and $\kappa = 0$ for a plane.

Often, the interior of the interface (surface) can be presented as $\{\mathbf{x} \in \Omega : H(\phi(\mathbf{x})) = 1\}$, where $H(.)$ is the standard Heaviside function. Using this notation, the integration of some function f over the interior can be given by $\int_\Omega f\, H(\phi(\mathbf{x}))\, d\mathbf{x}$. The directional derivative of the Heaviside function $H(.)$ in the normal direction $\boldsymbol{n}$ is given by $\nabla H(\phi(\mathbf{x})).\boldsymbol{n} = \delta(\phi)|\nabla\phi|$, where $\delta(.)$ is the standard Dirac function on the real line ($\delta(u) = \partial H / \partial u$). As such, the integral of the function f over only the boundary Γ can be presented by $\int_\Omega f\ \delta(\phi(\mathbf{x}))|\nabla\phi(\mathbf{x})|\, d\mathbf{x}$.

B. Multi-view Stereo Level-set Formulation

The object to be reconstructed is to be represented by a level set function ϕ . We seek the zero level set of ϕ that minimizes the energy functional

$$E(\phi) = \int_\Omega \mathbf{\Phi}(\mathbf{X})\ \delta(\phi)|\nabla\phi|\ dXdYdZ + \mu\int_\Omega \delta\ (\phi)|\nabla\phi|\ dXdYdZ\,, \quad (6)$$

where $\mathbf{\Phi}(\mathbf{X})$ is some matching score for all points $\mathbf{X} = (X,\ Y,\ Z) \in \Omega \subset R^3$, which should be minimized on *only* the object surface (boundary). The integral (6) is however done on the whole 3D space Ω by introducing the term $\delta(\phi)|\nabla\phi|$ in the integral. The second term is a smoothness term weighted in the functional by the regularizing parameter μ that to be chosen a priori. One advantage of this formulation is that one can easily model any available information on the object shape into the energy functional (6); for example we can add a priori information to force the level set function to reconstruct a predefined object.

The function $\mathbf{\Phi}(\mathbf{X})$ is based on the normalized cross-correlation and taken as

$$\mathbf{\Phi}(\mathbf{X}) = \sum_{\substack{i,j=1 \\ i \neq j}}^{n_v} \mathbf{\Phi}_{ij}(\mathbf{X}) \quad (7)$$

where n_v is the number of visible cameras to the current voxel $\mathbf{X}$ and the cross-correlation, $\mathbf{\Phi}_{ij}(\mathbf{X})$, between two visible cameras i and j is given by [18]

$$\mathbf{\Phi}_{ij}(\mathbf{X}) = 1 - \frac{< I_i, I_j >}{\sqrt{< I_i, I_i > \cdot < I_j, I_j >}}. \quad (8)$$

As such, similar to [8], the functional (6) works best when the object surface is textured. The correlation (8) is computed over two fixed windows in the two images I_i, I_j . The window in the image I_i is taken around the pixel coordinates (r_i, c_i) of the projection of the scene point $\mathbf{X}$ onto the image I_i via the projection matrix $\mathbf{P}_i$,

$$(r_i, c_i) = \boldsymbol{\pi}(\tilde{\mathbf{x}}_i) = \boldsymbol{\pi}(\mathbf{P}_i\tilde{\mathbf{X}}) = \boldsymbol{\pi}(u_i, v_i, w_i) = \left(\frac{u_i}{w_i}, \frac{v_i}{w_i}\right), \quad (9)$$

where the *inhomogenizing* transformation $\boldsymbol{\pi}$ (converts form homogenous to inhomogeneous coordinates). The quantities in (8) are thus given by

$$< I_i, I_i >= \sum_{m=-\frac{h}{2}}^{\frac{h}{2}} \left(\sum_{n=-\frac{w}{2}}^{\frac{w}{2}} \left(I_i\ r_i + m, c_i + n) - \overline{I}_i\right)^2, \quad (10)$$

$$< I_j, I_j >= \sum_{m=-\frac{h}{2}}^{\frac{h}{2}} \left(\sum_{n=-\frac{w}{2}}^{\frac{w}{2}} \left(I_j\ r_j + m, c_j + n) - \overline{I}_j\right)^2, \quad (11)$$

$$< I_i, I_j >= \sum_{m=-\frac{h}{2}}^{\frac{h}{2}}, \sum_{n=-\frac{w}{2}}^{\frac{w}{2}} \left(I_i\left(r_i + m\ c_i + n\right) - \overline{I}_i\right) \times \left(I_j(r_j + m, c_j + n) - \overline{I}_j\right), \quad (12)$$

where h and w are the height and width of the correlation window. We take h =5, and w =5 in all our experiments. The quantities $\overline{I}_i$ and $\overline{I}_j$ denote the mean values of I_i and I_j respectively

$$\overline{I}_i = \frac{1}{h \times w} \sum_{m=-\frac{h}{2}}^{\frac{h}{h}}, \sum_{n=-\frac{w}{2}}^{\frac{w}{2}} I_i\left(r_i + m\ c_i + n\right), \quad (13)$$

$$\overline{I}_j = \frac{1}{h \times w} \sum_{m=-\frac{h}{2}}^{\frac{h}{h}}, \sum_{n=\frac{w}{2}}^{\frac{w}{2}} I_j\left(r_j + m\ c_j + n\right). \quad (14)$$

Summation that appears in (7) is computed only for those points of the surface $S = \{\mathbf{X} : \phi(\mathbf{X}) = 0\}$ which are visible in

the two concerned images. Thus, estimating $\mathbf{\Phi}(\mathbf{X})$ requires computing the hidden parts of the surface for all cameras.

C. Visibility

At each step, in order to compute the summation in (7), we need to compute the visibility, χ. It is equivalent to the problem of determining which part of the surface is visible from a given view point (the camera center in our case). This is a classical problem in computer graphics. A typical approach to this problem is the so-called ray tracing. The idea is to start from each point in the domain of interest, shoot a ray towards the view point, and check the number of times this ray hits the surface. Unfortunately this intuitive algorithm turns out to be computationally expensive. However, it is possible to exploit the level-set representation of surfaces to efficiently solve the problem. We adopt here a level-set implementation of the implicit ray tracing technique that is originally reported in [19]. This is a one-pass algorithm that finds the line of sight for a given configuration of implicit surfaces in an incremental way. The algorithm computes another level set function ψ, which tells us the portions of ϕ that are visible from the view point. More precisely, $\{\mathbf{X} \in \Omega : \psi(\mathbf{X}) \geq 0\}$ will be the regions visible from a view point $\mathbf{v}$, see Figure 3. Therefore, the desired visibility function can be written as $\chi = H(\psi)$. For more detail on implicit ray tracing, we redirect readers to [19].

D. Evolution Equation

Now we can rewrite (7) to reflect the visibility as:

$$\mathbf{\Phi} = \sum_{\substack{i,j=1 \\ i \neq j}}^{n} \chi(i,\mathbf{X})\chi(j,\mathbf{X})\mathbf{\Phi}_{ij}, \tag{15}$$

where n is the number of all views, $\chi(z,\mathbf{X})$ is a characteristic function which denotes the visibility of the voxel $\mathbf{X}$ to the camera z

$$\begin{cases} \chi(z,\boldsymbol{X}) = 1 & \text{if } \mathbf{X} \text{ is visible to camera z,} \\ \chi(z,\boldsymbol{X}) = 0 & \text{if } \mathbf{X} \text{ is not visible to camera z.} \end{cases} \tag{16}$$

The Euler-Lagrange equation for ϕ of the functional (6) can be shown (after some lengthy computation) equal to

$$\frac{\partial E}{\partial \phi} = -\delta(\phi)\left(\nabla\mathbf{\Phi}\cdot\boldsymbol{n} + \mathbf{\Phi}\kappa + \mu\kappa\right), \tag{17}$$

where κ is the mean curvature. Employing an artificial time $t > 0$, the evolution equation becomes

$$\frac{\partial \phi}{\partial t} = -\frac{\partial E}{\partial \phi} = \delta(\phi)\left(\nabla\mathbf{\Phi}\cdot\boldsymbol{n} + \mathbf{\Phi}\kappa + \mu\kappa\right). \tag{18}$$

Note here, on the contrary to other approaches [8][18] we neither drop any terms nor approximate the other terms; all the terms of the resulting evolution equation are used.

In our implementation, we use a regularized form of $\delta(\phi)$ [20]

$$\delta_{\varepsilon}(\phi) = \frac{1}{\pi}\,\frac{\varepsilon}{\varepsilon^2 + \phi^2}\ . \tag{19}$$

This regularized form $\delta_{\varepsilon}(\phi)$ is used in (18) in place of $\delta(\phi)$. Using this approximation, the algorithm has the tendency to compute a global minimizer. One of the reasons is that, the Euler-Lagrange equation acts only locally, on a few level surfaces around $\phi = 0$ using the original Dirac function, while by the regularized form, the equation acts on all level sets, of course stronger on the zero level set, but not only locally. In this way, in practice, we can obtain a global minimizer, independently of the position of the initial set. The final evolution equation is hence given by

$$\frac{\partial \phi}{\partial t} = -\frac{\partial E}{\partial \phi} = \delta_{\varepsilon}(\phi)\left(\nabla\mathbf{\Phi}\cdot\boldsymbol{n} + \mathbf{\Phi}\kappa + \mu\kappa\right). \tag{20}$$

We now turn to some implementation issues. The term $\nabla\mathbf{\Phi} = \dfrac{\partial\mathbf{\Phi}}{\partial\mathbf{X}}$ is given by

$$\nabla\mathbf{\Phi} = \sum_{\substack{i,j=1 \\ i \neq j}}^{n} \chi(i,\mathbf{X})\chi(j,\mathbf{X})\nabla\mathbf{\Phi}_{ij}. \tag{21}$$

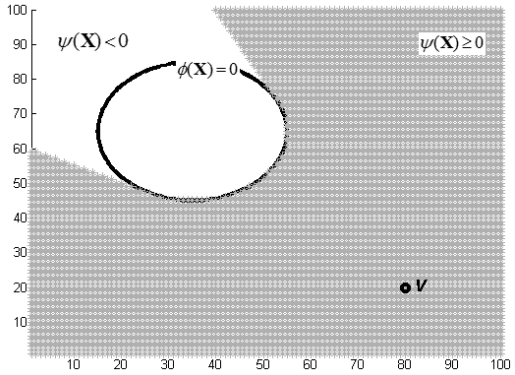

Figure 3. Illustration of visibility algorithm in 2D. The view point v is visible to all points in the region $\psi(\mathbf{X}) \geq 0$, the gray region, and invisible to the region $\psi(\mathbf{X}) < 0$, the white region.

The gradient $\nabla\mathbf{\Phi}_{ij}$ can be computed directly from (8). However we will need to calculate the following derivatives: $\dfrac{\partial I_i}{\partial\mathbf{X}}$ and $\dfrac{\partial I_j}{\partial\mathbf{X}}$, which can proceed as follows. Using the chain rule we have

$$\frac{\partial}{\partial\mathbf{X}} I_i\left(\boldsymbol{\pi}\left(\mathbf{P}_i\tilde{\mathbf{X}}\right)\right) = \frac{\partial I_i(\mathbf{P}_i\tilde{\mathbf{X}})^T}{\partial\boldsymbol{\pi}} \cdot \frac{\partial\boldsymbol{\pi}\left(\tilde{\mathbf{x}}_i\right)}{\partial\tilde{\mathbf{x}}_i} \cdot \frac{\partial\tilde{\mathbf{x}}_i}{\partial\tilde{\mathbf{X}}} \cdot \frac{\partial\tilde{\mathbf{X}}}{\partial\mathbf{X}} \tag{22}$$

where

$$\frac{\partial I_i(\mathbf{P}_i\tilde{\mathbf{X}})^T}{\partial\boldsymbol{\pi}} = \nabla I_i, \tag{23}$$

and

$$\frac{\partial\boldsymbol{\pi}(\tilde{\mathbf{x}}_i)}{\partial\tilde{\mathbf{x}}_i} = \frac{\partial(u_i/w_i, v_i/w_i)}{\partial(u_i, v_i, w_i)} = \begin{bmatrix} 1/w_i & 0 & -u_i/w_i^2 \\ 0 & 1/w_i & -v_i/w_i^2 \end{bmatrix}. \tag{24}$$

And two last ingredients needed,

$$\frac{\partial \tilde{\mathbf{x}}_i}{\partial \tilde{\mathbf{X}}} = \mathbf{P}_i. \quad \text{and} \quad \frac{\partial \tilde{\mathbf{X}}}{\partial \mathbf{X}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}. \qquad (25)$$

As such, we have all pieces need to compute $\frac{\partial I_i}{\partial \mathbf{X}}$ and $\frac{\partial I_j}{\partial \mathbf{X}}$, which allows us to calculate $\nabla \mathbf{\Phi}_{ij}$ then $\nabla \mathbf{\Phi}$.

The numerical implementation of the PDE evolving (20) is carried out on the discretized Cartesian grid using the semi-implicit scheme [20][23] to satisfy the Courant-Friedrichs-Lewy (CFL) condition [23]. By using this scheme we can speed up the evolution of the level set function.

Implementation note: The background of the object can be segmented out from the input images beforehand. So the background pixels in each image can be given a specific color (e.g., black). A better scenario though is to mark those background pixels with alternating colors in the input images (e.g., white in one image and black in another, and so on). This will cause those points to have very little correlation among the various views (thus contributing significantly to the error criterion (6)). Consequently, this will *softly* guide the level set evolution to exclude from the object the 3D points being projected to background pixels in any of the input images. This scenario improves further the results of the approach in a straightforward manner, without the need for any modification in the level set evolution, and more importantly, without the need for taking hard decisions on those points, the case that may lead to unrecoverable reconstruction errors.

IV. Experimental Results

In this section, the proposed approach is evaluated extensively using several experiments on different datasets. Firstly the approach (implemented in Matlab, but the data acquisition program with the GUI implemented in Visual C++), is applied to a synthetic dataset generated using the AutoCAD program. This experiment using the AutoCAD environment helps quantify the performance versus ground-truth results under varying degrees of artificial noise. Then the practical usefulness of the proposed approach is demonstrated through its application to real datasets publically available from the internet, as well as real datasets obtained using the system setup which we have constructed.

A. Synthetic Dataset

We apply our approach to a synthetic dataset generated using the AutoCAD 2007 program. We use AutoCAD program to simulate the developed system. As described in section II, the camera must be calibrated at the initial view. In this experiment, we accomplish this using a synthetic calibration pattern, see **Error! Reference source not found.**4.

The captured image is then used to find $\mathbf{P}_0$ as explained in Section II, from which all the other projection matrices are derived given the rotation angle of the object (rotary table). Then the setup is used to get 12 images of a horse object by rotating the object about the Y axis and taking an image every 30º, see Figure 5. The proposed approach is applied to those images, and some rendered views of the reconstructed object are shown in Figure 6. Apart from some voxelization effect due to the numerical implementation on a discretized grid, the horse shape is accurately modeled. Due to our efficient shape representation, notice how the fine details of the horse's legs and tail are correctly reconstructed. The size of volume in this experiment is 160×160×160 voxels, and it takes about 65 minutes to complete on a P4 3GHz PC with 2Gbytes RAM. The initial level set is a sphere with radius 0.01.

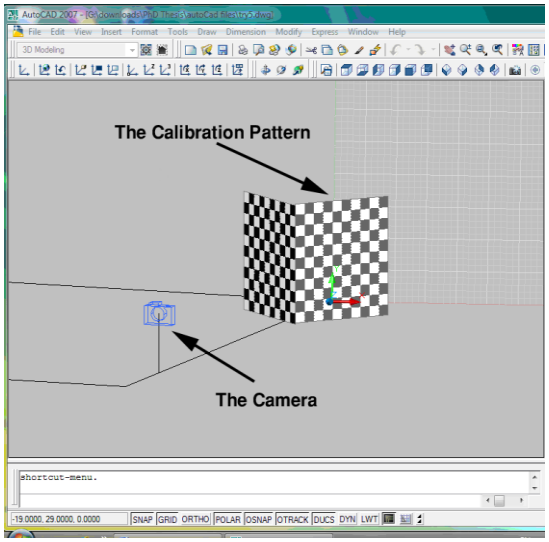

Figure 4. Using AutoCAD to simulate our developed setup and acquire the image of the calibration pattern.

To simulate non-ideal conditions in real environments, input images are noised by additive Gaussian noise with zero mean and standard deviation, σ, that is varied from 0 to 70 in steps of 10. Then at each value of σ, our approach is applied to the input images. To assess the quality of the reconstructed shape, the 3D reconstruction from the noisy images is re-projected onto the different view directions and the silhouettes are obtained as illustrated in Figure 7**.** As an accuracy measure, we use the root mean square error (E_{RMS}) between those obtained silhouettes and the corresponding ground-truth (noise-free) silhouettes:

$$E_{RMS} = \sqrt{\frac{1}{n_I H_I W_I} \sum_{i=1}^{n_I} (I_i^{proj} - I_i^{gt})^2}, \qquad (26)$$

where n_I is the number of images, H_I is the image height, W_I is the image width, I_i^{proj} is the projected image to the *i*-th view, and I_i^{gt} is the corresponding ground-truth noise-free image for the *i*-th view. For the sake of comparison, the same procedure is repeated using Faugeras and Keriven's approach [8]. The plot of E_{RMS} versus for the two approaches is shown in Figure 8. From this figure, one can notice consistently the better and robust performance of the proposed method over the approach in [8] versus all levels of noise. Up to the high noise level of 50, E_{RMS} remains below

0.18 for our approach, while the other approach sooner exceeds this E_{RMS} level. For noise higher than that, the input images hardly show the details of the horse, and accordingly E_{RMS} starts to increase notably. However, as shown in Figure 7(d) and (e), even in this case, the silhouettes of the reconstructed horse are recovered with rather good accuracy.

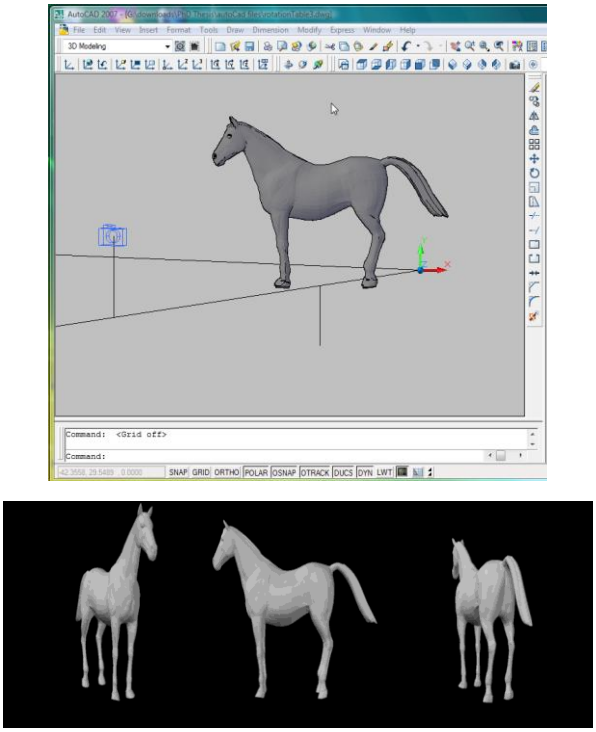

Figure 5. Using AutoCAD to simulate our developed setup and acquire images from multi-views of an object. (Top) Horse. (Bottom) Some views of 12-frame horse sequence taken using AutoCAD simulated camera.

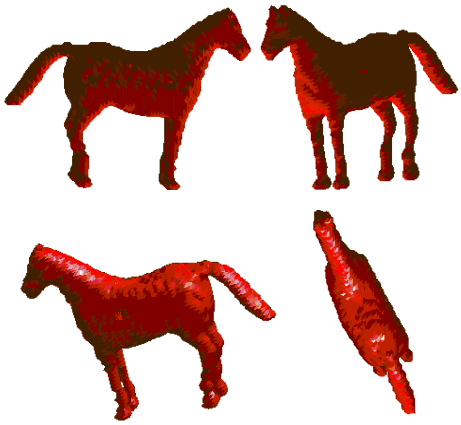

Figure 6. Some rendered views of the final reconstructed 3D model by the proposed approach.

Note also that in our experiment with the approach in [8], care is taken to initialize it properly as it is sensitive to where the evolution has started, which is not the case with our approach.

B. Real Datasets from the Internet

To evaluate our approach on real objects we apply it first to reconstruct objects whose datasets are available on the internet. Each dataset contains the projection matrices of each view. The backgrounds of the downloaded images are segmented manually.

Figure 9 shows some images of a 12-frame sequence for a rooster dataset obtained from the Computer Vision and Image Processing Lab at the University of Louisville [33]. To run our approach, the initial zero level set function was taken as a sphere, where several locations and sizes have been experimented with. Figure 10(a) illustrates some of those initial zero level sets along with the reconstructed 3D rooster model. The proposed approach was able to obtain very good reconstruction results starting from various initial level sets. Notice also how the rooster's crown that has sharp and thin parts is accurately reconstructed. You can compare the results in Figure 10(a) with the results of the space carving approach in Figure 10(b), which showed a noisy result with missing parts and several floating voxels. Our results here are smoother and there are no floating voxels in the obtained 3D model. Some evolution stages are shown in Figure 11. In this experiment we used a volume of size 80×80×80, and μ =100. On a P4 with speed 2.8 GHz PC with 1GBytes of RAM; it takes about 30 minutes to reach the final shape.

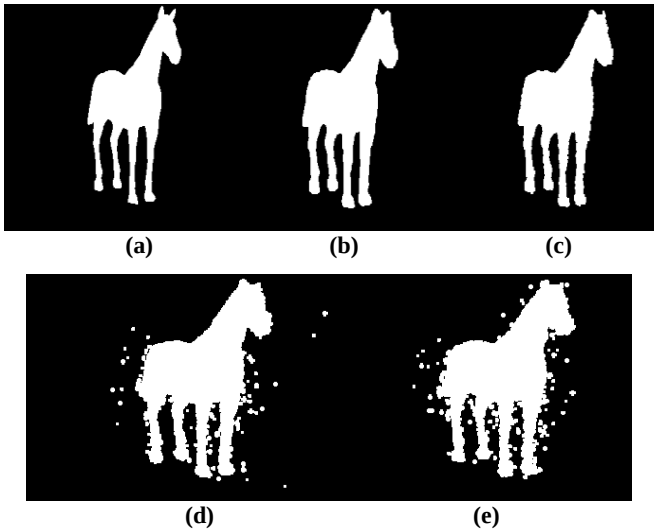

Figure 7. The ground-truth (noise-free) silhouette of the first view of the horse sequence in (a) versus the silhouettes of the projections of the reconstructed shape onto the first view at different values of σ : (b) σ =0, (c) σ =50, (d) σ =60, (e) σ =70.

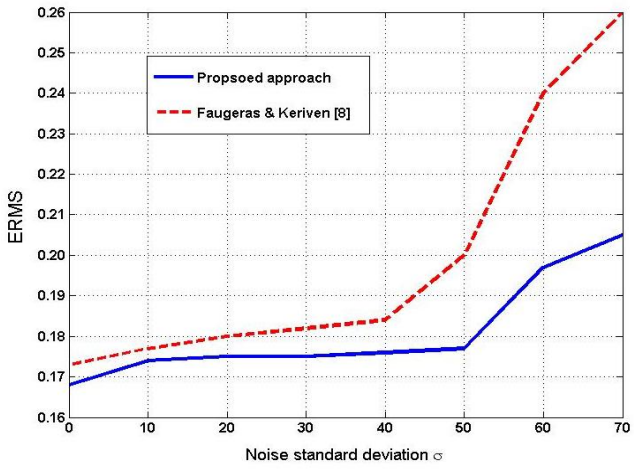

Figure 8. E_{RMS} versus the noise standard deviation σ for the proposed approach and Faugeras and Keriven's approach [8] on the horse sequence.

Our method is also applied to the Oxford dinosaur dataset [31]. Figure 12 shows some of the 36 images used. Figure 13 shows the results of applying our method starting from different initial level set functions. As shown in this figure our approach could reconstruct the object regardless of the position and size of the initial level set function. The dinosaur's hands, feet and tail are correctly reconstructed in very good details. The volume size used in this experiment is 140×140×140 and μ =100.

Figure 9. Some views of 12-frame rooster sequence.

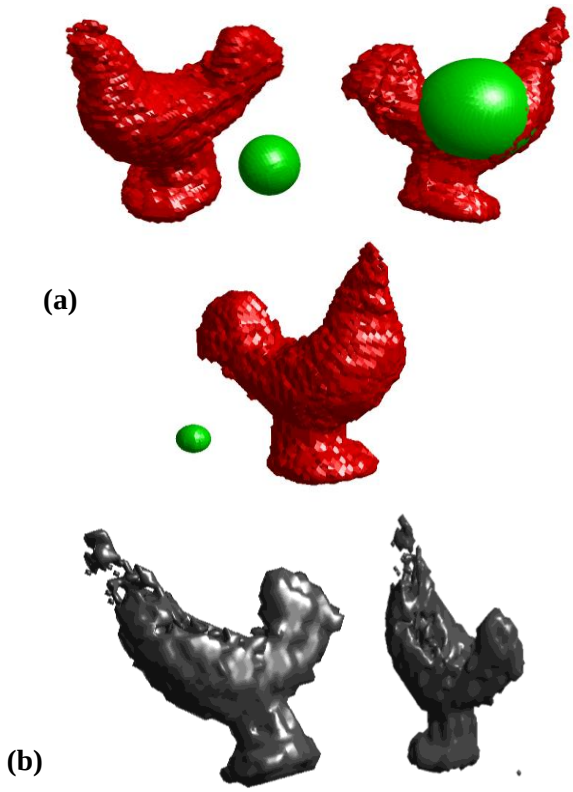

Figure 10. (a) Some initial level set functions (spheres shown in green) and the final reconstructed 3D model (shown in red) by the proposed approach. (b) Two rendered views of the reconstruction by the space carving technique [14] using the same input images.

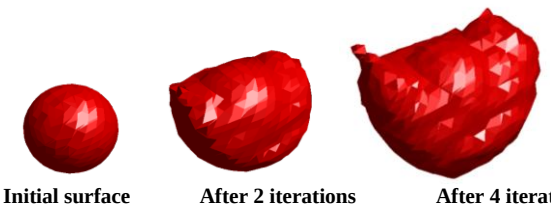

Figure 11. The evolution of the level set function to reconstruct the final surface for the experiment on the 12-frame rooster sequence.

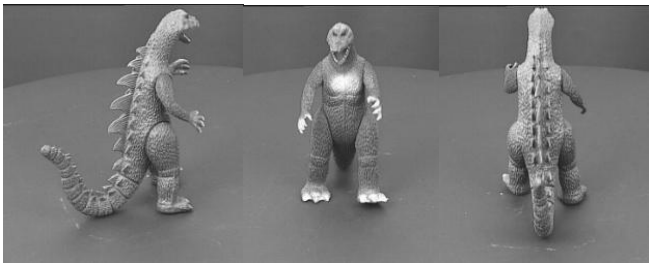

Figure 12.Some views of 36-frame dinosaur sequence.

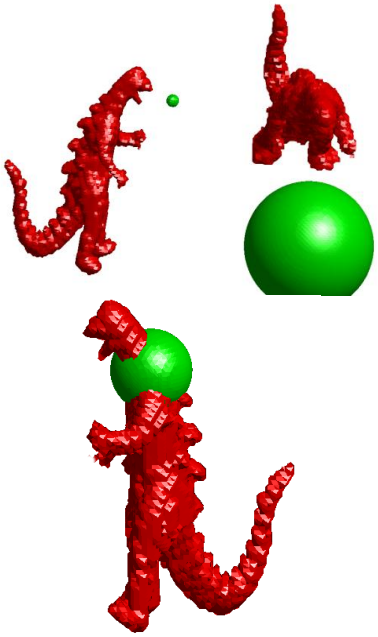

Figure 13. Several different initial level sets (spheres shown in green) and the final 3D surface (shown in red) for the dinosaur sequence.

Our approach is also applied to a sequence of 24 images for another dinosaur [32]. **Error! Reference source not found.** shows some of these images. In Figure 14, we see the result of applying this method, as before, starting from different initializations. As shown in figure, the fine details of dinosaur's fingers and tail are accurately modeled. The volume size used in this experiment is 100×100×100 and μ =100. It takes about 1 hour to get the final surface in 4 iterations.

C. Real Datasets using Our Setup

To evaluate the proposed approach using the developed setup, we apply it to some other objects. Figure 16 shows some of 12 images of a baby toy. The background in the input images has been segmented out manually. Figure 16 shows some views of the reconstructed object. In this experiment we used a volume of size 80×80×80, and μ =100. The approach needed four iterations to reach the final shape taking about 30 minutes on a

P4 with speed 2.8 GHz with 1GBytes of RAM. Another experiment is done on a duck toy for which, 12 images are taken by our setup, see Figure 17. Figure 18 shows some views of the reconstructed object. In this experiment we used a volume of size 80×80×80, and μ =100. The four iterations needed to reach the final shape took about 20 minutes.

Figure 14.Some views of 24 images of dinosaur #2

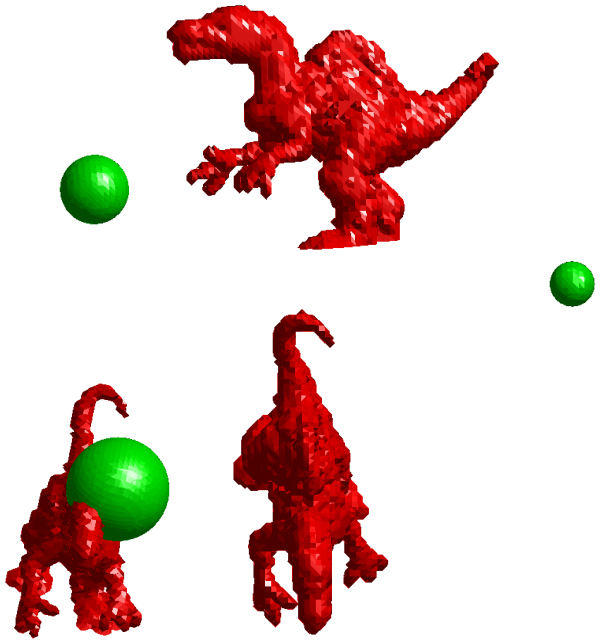

Figure 14. Several initial level set functions (spheres shown in green) and some views of the final reconstructed 3D model (shown in red) for dinosaur #2.

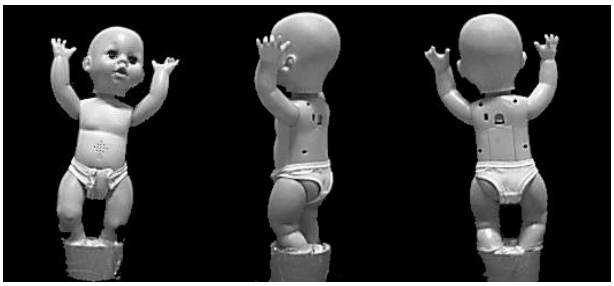

Figure 15. Some images of a 12-frame sequence of a baby toy taken by our setup.

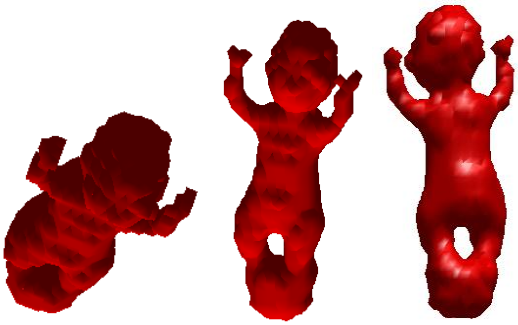

Figure 16. Some views of the final reconstructed baby toy.

Figure 17. Some images of a 12-frame sequence of a duck toy taken by our setup.

V. CONCLUSIONS

We have presented a simple, yet effective system for complete multi-view 3D shape reconstruction consisting of a rotary-table and a USB camera, both controlled via a desktop PC. The system offers a cost-effective solution to the multi-view stereo acquisition problem without the need for using several cameras. A variational approach has been formulated and developed to reconstruct the 3D object shape from the acquired sequence of calibrated images. In contrast to existing methods, this approach presents a flexible shape representation and an efficient optimization algorithm with no special initialization requirement. The object is represented as a level set from the first problem formulation. This also allows the easy incorporation of any available shape a priori information in the energy functional in order to guide the surface evolution. Our extensive experimental results have shown the proposed approach can successfully find the 3D shape regardless of the position of the initial surface. Results have also demonstrated that fine details of the objects have been correctly recovered.

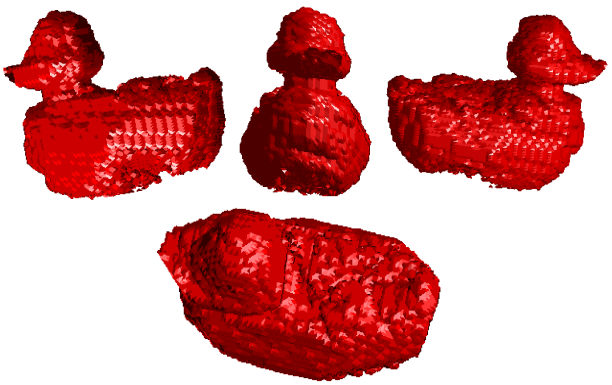

Figure 18. Some views of the final reconstructed duck toy.

Our current efforts are directed to utilize the ease of incorporating prior shape information in our approach in order to reconstruct objects with shape variations consistent with a set of training model examples. This can offer a great

advantage when working with a specific category of objects. Some early implementation and results of our idea are already drafted in [24]. Another direction for our current research efforts is to address the time performance of the approach. The developed approach takes a small number of iterations (typically 4-8) to converge, but a single iteration may take rather a long time (about 3-12 minutes depending on the volume size) due to the high computational cost for the various approach operations (e.g., visibility calculation and level set evolution). One possibility to reduce this time is converting our shape reconstruction code from Matlab to a fully compiled programming language (e.g., C++). Another important possibility is to utilize parallel programming concepts to carry out computations concurrently on multi-core CPUs available nowadays on desktop PCs.

REFERENCES

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms", Int. J. of Computer Vision, 47(1-3), pp. 7–42, 2002.

[2] Soon-Yong Park and Murali Subbarao, "A multiview 3D modeling system based on stereo vision techniques", Machine Vision and Applications, Vol. 16, pp. 148–156, Dec.2005.

[3] S. Osher and J. Sethian, "Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations," Journal of Computational Physics, vol. 79, pp. 12–49, 1988.

[4] S. Seitz, B. Curless, J. Diebel, D. Scharstein and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in IEEE International Conference on Computer Vision and Pattern Recognition(CVPR), June 2006, pp. 519-526.

[5] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 8, pp. 993–1008, January 2003.

[6] Kalin Kolev, Maria Klodt, Thomas Brox and Daniel Cremers, "Continuous Global Optimization in Multiview 3D Reconstruction, " Int. Journal of Computer Vision, Volume 84, Number 1, 80-96, 2009.

[7] Kiriakos N. Kutulakos and Steven M. Seitz, "A theory of shape by space carving," Int. Journal of Computer Vision, 38(3), pp. 199-218, 2000.

[8] O. Faugeras and R. Keriven, "Variational principles, surface evolution, PDEs, level set methods, and the stereo problem," IEEE Transactions on Image Processing 7(3), pp. 336-344, 1998.

[9] R. I. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge University Press, 2003.

[10] L. Robert, "Camera calibration without feature extraction," Computer Vision and Image Understanding (CVIU), 63(2), pp. 314-325, 1996.

[11] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: active contour models," Int. Journal of Computer Vision, vol. 1, pp. 321–331, 1987.

[12] Aldo Laurentini, "The visual hull concept for silhouette-based image understanding," IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(2), pp. 150-162, 1994.

[13] John Isidoro and Stan Sclaroff, "Stochastic refinement of the visual hull to satisfy photometric and silhouette consistency constraints," in International Conference on Computer Vision, 2003, pp. 1335-1342.

[14] Kiriakos N. Kutulakos and Steven M. Seitz, "A theory of shape by space carving," Int. Journal of Computer Vision, 38(3), pp.199-218, 2000.

[15] A. Broadhurst, T. Drummond, and R. Cipolla, "A Probabilistic framework for space carving," in IEEE International Conference on Computer Vision, 2001, pp. 388-393.

[16] R. Yang, M. Pollefeys, and G. Welch, "Dealing with textureless regions and specular highlights: a progressive space carving scheme using a novel photo-consistency measure," in IEEE Int. Conf. on Computer Vision, 2003, pp. 576-584.

[17] A. Yezzi and S. Soatto, "Stereoscopic segmentation," International Journal of Computer Vision 53(1), pp. 31-43, 2003.

[18] Hailin Jin, "Variational methods for shape reconstruction in computer vision," PhD thesis, Elect. Eng. Dept., Washington University, 2003.

[19] Y. H. Tsai, L.-T. Cheng, S. Osher, P. Burchard, G. Sapiro," Visibility and its dynamics in a PDE based implicit framework," in Journal of Computational Physics 199, pp. 260–290, 2004.

[20] L. Vese and T. Chan, "A multiphase level set framework for image segmentation using the Mumford and Shah model," in International Journal of Computer Vision, vol. 50(3), pp. 271-293, 2002.

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

[22] Moumen El-Melegy and Nagi Al-Ashwal, "Lens distortion calibration using level sets," lecture notes in computer science, N. Paragios et al. (Eds.)., Springer-Verlag, Berlin, LNCS 3752, 2005, pp. 356 - 367.

[23] A. R Forsyth, Calculus of Variations, Dover Publications, New York, 1960.

[24] Moumen El-Melegy, Nagi Al-Ashwal and Aly Farag, "Model-based multiview stereo via level sets with statistical shape prior," in IEEE Int. Conf. on Image Processing (ICIP'11), Belgium, Sept. 11-14, 2011.

[25] Aly Abdelrahim, Moumen El-Melegy, and Aly Farag, "Realistic 3D Reconstruction of the Human Teeth using Shape from Shading with Shape Priors," in IEEE workshop on Medical Computer Vision, associated CVPR'12, Rhode Island, June 16-21, 2012.

[26] Carlos Leung, Ben Appleton, Mitchell Buckley and Changming Sun, "Embedded Voxel Colouring with Adaptive Threshold Selection Using Globally Minimal Surfaces," Int. J. of Computer Vision, pp.1-17, 2012.

[27] H. Vu, P. Labatut, J. Pons, and R. Keriven, "High Accuracy and Visibility-Consistent Dense Multiview Stereo," IEEE Trans. Pattern Analysis and Machine Intelligence, vol.34, no.5, pp.889-901, 2012.

[28] D. Cremers, and K. Kolev, "Multiview Stereo and Silhouette Consistency via Convex Functionals over Convex Domains," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.33, no.6, pp.1161-1174, 2011.

[29] Liuxin Zhang and Yunde Jia, "Surface reconstruction from Images using a variational Formulation," Advances in Multimedia Modeling, Lecture Notes in Computer Science, Volume 5916, 2010, pp. 4-14.

[30] G. Vogiatzis, C. Hernandez, P. Torr, and R. Cipolla, "Multiview Stereo via Volumetric Graph-Cuts and Occlusion Robust Photo-Consistency," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.29, no.12, pp.2241-2246, 2007.

[31] Visual Geometry Group, [online], http://www.robots.ox.ac.uk/~vgg/data/data-view.html (Accessed: 2 July, 2012).

[32] 3D Photography Dataset, [online], http://www.cs.washington.edu/homes/furukawa/research/mview/index.html (Accessed: 4 July, 2012).

[33] CVIP Lab, http://www.cvip.louisville.edu/ (Personal communication)