Abstract—Most of the previous multi-view reconstruction algorithms focus on minimizing reconstruction distortion, i.e. reconstructing a 3D model as close to the real object as possible, and rely on subsequent simplification process to control the reconstruction rate for practical usage. In this paper, in addition to reconstruction distortion, we directly consider the reconstruction rate in wide-baseline multi-view reconstruction. In particular, we develop a novel rate-distortion efficient multi-view reconstruction system, which consists of two major components: the PDE-based mesh optimization and the local subdivision based mesh refinement. Although both components are prior arts, to achieve rate-distortion optimization, new ingredients are proposed including non-uniform sample distribution for efficient and accurate surface distortion approximation in mesh optimization and non-uniform vertex distribution in mesh refinement. From a set of calibrated and segmented wide baseline images, our developed system is able to reconstruct the optimal 3D mesh that achieves minimum reconstruction distortion at any rate within a certain range. Experimental results show that our proposed system significantly outperforms the original PDE-based reconstruction method in both rate-distortion performance and visual quality.

Index Terms—3D reconstruction, multi-view reconstruction, rate-distortion optimization, PDE based mesh optimization.

I. INTRODUCTION

With the increasing demands on 3D applications and the easy capturing of 2D images nowadays, building 3D models from 2D images receives much attention in the past few years. Based on the camera settings, 3D reconstruction from multi-view images can be categorized into two main groups: small baseline reconstruction and wide baseline reconstruction. The cameras in small baseline reconstruction are placed close to each other so as to obtain the stereo information. In such a setting, the number of cameras needed is typically large in order to capture the entire scene. In contrast, the number of cameras needed for wide baseline reconstruction is much less at the cost of placing cameras farther apart, where stereo information is not available.

The quality of reconstructed 3D models can be evaluated based on two main criteria: reconstruction distortion and reconstruction rate. Reconstruction distortion tells how much the difference is between a reconstructed model and the corresponding real-world 3D object, while reconstruction rate conveys how compact the representation of a reconstruction model is.

Reconstruction distortion has been the main issue in most of the previous multi-view reconstruction techniques. We refer readers to the nice surveys by Seitz et al. [1] and Dyer [2]. Most of the previous multi-view reconstruction algorithms focus on how to reconstruct the 3D model as close to the real object as possible. In other words, little attention has been paid to the issue of reconstruction rate. Reconstruction algorithms generated by the state-of-the-art multi-view stereo algorithms [1] or obtained from 3D scanners [3] are typically of huge volume, which are not suitable to be directly used in practical applications. Simplification methods such as [4] need to be applied as a post-processing to reduce the size of the reconstructed models so that they can become usable.

Such a fine-to-coarse processing, i.e. high quality multi-view reconstruction followed by simplification, is resource-inefficient since it over-creates numerous vertices or 3D points at the beginning and then removes many of them in the simplification stage. Here comes out the question: can we directly consider the reconstruction rate in the multi-view reconstruction instead of relying on the subsequent simplification to control the rate? Moreover, considering that different applications have different rate requirements, it is highly desired to have a multi-rate 3D reconstruction, i.e. we only need to reconstruct the 3D model once and the representation can be used to fit different rate requirements with different reconstruction quality. This progressive feature is very useful for the applications of progressive transmission and rendering.

In this paper, we propose a framework for rate-distortion optimized progressive 3D reconstruction from a set of calibrated and segmented wide baseline images. In particular, a coarse triangular mesh is first built up and then optimized by using the PDE-based mesh optimization method [5], [6], which minimizes the distortion at a certain rate. We then keep applying a local subdivision approach to add new vertices to the most distorted triangle until reaching the target reconstruction rate. For each local subdivision, the newly added vertices and their adjacent vertices are optimized using the same PDE-based optimization method. Note that although the major components in the developed system are prior arts, to achieve rate-distortion optimization, new ingredients are proposed including non-uniform sample distribution for efficient and accurate surface distortion approximation in mesh optimization and non-uniform vertex distribution in mesh refinement. From a set of calibrated and segmented wide baseline images, our developed system is able to reconstruct the optimal 3D mesh that achieves minimum reconstruction distortion at any rate.
within a certain range. Experimental results show that our proposed system significantly outperforms the original PDE-based reconstruction method [6].

We would like to point out that a recent work by Imre et al. [7] has already explicitly addressed the rate-distortion issue in their 2D-to-3D reconstruction problem. However, their work is limited to a set of two stereo images and focuses only on stereo-based scene reconstruction, which is totally different from our problem. The multi-rate multi-view reconstruction has also been implicitly mentioned in [5], [6], which suggest to apply local subdivision to refine high-curvature mesh regions. However, their assumption is that the initial mesh is already a good-quality one. If the initial mesh is as coarse as that in our case, which cannot approximate high-curvature regions well, their suggested method will fail.

The main contributions of this paper lie in three aspects. (1) We introduce the rate-distortion concepts and computations into wide baseline multi-view reconstruction. To the best of our knowledge, for wide baseline multi-view reconstruction, there is no prior work addressing similar issues. (2) We improve the sampling method used in the PDE-based optimization so as to achieve accurate surface distortion approximation with low complexity. (3) We propose a novel rate-distortion optimized refinement algorithm that allocates more vertices to the regions with more details.

The rest of the paper is organized as follows. We give an overview of the developed multi-view reconstruction system in Section II. The three major steps in our system, i.e. initial mesh generation, mesh optimization and mesh refinement, are described in detail in Section III, IV and V, respectively. Finally, we show the experimental results in Section VI and conclude the paper in Section VII.

II. SYSTEM OVERVIEW

Our work addresses the problem of wide baseline reconstruction. The primary inputs of our system are a set of \( n \) segmented color images \( I = \{I_i|i = 1 \cdots n\} \) and a set of associated projection matrices \( P_i \). The cameras are placed in a circle and the 3D object is at the center. All the cameras are calibrated to obtain the projection matrices when capturing the input images. In addition, we assume that the surface to be reconstructed is a Lambertian surface, i.e. the brightness of the surface is independent of the observer’s angle of view, which is a common assumption for most algorithms [1]. The output of our system is the reconstructed 3D object represented by triangular mesh. The mesh representation is a very popular format since it facilitates efficient computations on many local properties.

Our objective is to deliver a rate-distortion optimized progressive 3D reconstruction. Specifically, given a certain reconstruction rate, we want to achieve the minimum reconstruction distortion. On the other hand, given a certain reconstruction distortion, we want to find the smallest reconstruction rate. Similar to [7], we define the reconstruction rate as the number of vertices used for the reconstructed 3D model. The definition of the distortion will be discussed in detail in Section IV-B.

---

III. INITIAL MESH GENERATION

Our system consists of the following three major steps:

- **Step 1: Initial mesh generation.** Use the method in [8] to reconstruct a visual hull from a set of segmented images together with the projection matrices. The visual hull is then triangulated to obtain a surface representation by using Poisson surface approach [9].
- **Step 2: PDE-based mesh optimization.** Optimize the input mesh using the PDE-based surface flow technique that is similar to [5], [6], [10]. The output of the mesh optimization is an optimized mesh with the same number of vertices as the input. The mesh is optimal in the sense that it achieves the minimal distortion at the given rate.
- **Step 3: Progressive mesh refinement using local adaptive subdivision.** Repeatedly use local subdivision to add new vertices to the most distorted triangle if the target rate has not been met. After each refinement, go back to Step 2 to optimize the newly added vertices and their neighboring vertices.

The visual hull is then triangulated to obtain the the surface mesh representation using either the marching cube technique [11] or the Poisson surface approach [9]. We choose the Poisson surface approach as its triangulated surface is smoother. An example of a visual hull and the corresponding triangulated surface is shown in Fig. 1.

Because the triangulated surface obtained from the Poisson approach is irregular, a remeshing process [12] is further applied to obtain a regular surface representation. This regularization is necessary because the subsequent PDE-based surface optimization requires a roughly regular mesh to obtain stable
results. The size of the output regular mesh is set to a small one so as to have a wide range of distortion levels.

IV. MESH OPTIMIZATION USING PDE-BASED SURFACE FLOW

Our main goal is to reconstruct 3D model in a rate-distortion efficient way. In other words, given the initial coarse reconstruction with a small number of vertices, it is needed to optimize the positions of all the vertices in order to achieve the minimum geometry distortion at the current rate. Typically, global optimization is applied for such a surface optimization. In our work, we adopt the partial differential equation (PDE) based surface flow method [5], [6], [10] with our improved distortion measurement to optimize the coarse mesh. Note that the mesh optimization is applied not only to the initial coarse mesh but also to each of the subsequent refined meshes.

A. PDE-based Surface Reconstruction

The PDE-based surface flow is a gradient descent flow that is governed by the Euler-Lagrange equation with a certain underlying energy function. In our context, the energy function is the one that measures the photo-consistency distortion of all the reconstructed points on the surface [10], [13]:

$$E(S) = \sum_{p} G(p),$$ (1)

where $G(p)$ is a function that measures the photo-consistency distortion of point $p = (x, y, z)^T$ on the surface. The minimization of (1) produces an optimal surface in which all the surface points are photo-consistent to the input images. Specifically, each surface point should be photo-consistent among those images where it is visible.

To minimize the energy function, we need to solve the equation of $E'(S) = 0$, which is typically hard to solve directly for most computer vision applications. One general technique is to solve the PDE in a numerical way:

$$E'(S) = -\frac{\partial S}{\partial t}, \quad S(0) = S_0$$ (2)

where $S_0$ is the initial surface, and the variable $t$ is the time marching parameter. The solution is obtained by starting with an initial surface $S_0$ and then searching for a stable state of (2), i.e. $\frac{\partial S}{\partial t} = 0$.

Since the complete form of (2) is very complex, a simplified version is typically used [10]:

$$\frac{\partial S}{\partial t} = (G H + \langle \nabla G, \mathbf{N} \rangle) \mathbf{N}, \quad S(0) = S_0$$ (3)

where $H$ is the mean curvature, $G$ is the non-negative, monotonically decreasing function that measures the geometry distortion of the reconstructed surface, and the scalar $\langle \nabla G, \mathbf{N} \rangle$ is the derivation of $G$ along the normal vector $\mathbf{N}$. Basically, (3) drives each point on the surface to evolve along its current normal direction. The first term in (3) is a smoothing term for reducing the mean curvature of the surface, and the second term moves the surface in the direction of reducing the surface error.

Specifically, starting from a simple initial surface $S_0$, the surface is incrementally updated as

$$S(p)^t+\Delta t = S(p)^t + \frac{\partial S(p)^t}{\partial t} \Delta t,$$ (4)

where $\frac{\partial S(p)^t}{\partial t}$ is computed by (3) controls the evolving direction and the time step $\Delta t$ affects the evolution speed. Note that when updating the surface, the time step $\Delta t$ must satisfy the Courant-Friedrichs-Lewy stability criterion [14], i.e. the moving speed of each point on the surface must be strictly restrained by the minimum detail in the system.

B. Distortion Measurement

One important issue in the PDE system (3) is to define the function $G(p)$ that measures the photo-consistency distortion of point $p$ on the reconstructed surface. Since the reconstructed 3D surface is derived from multi-view images and our system operates in a coarse-to-fine manner, the best way to measure the photo-consistency distortion of a surface point is through measuring the color inconsistency of the projection image points in all the visible views. Particularly, we use a similar photo-consistency function as that in [15], i.e.

$$F(p) = \frac{1}{K-1} \left( \sum_{i=1}^{K} C_i^2(p) - \frac{1}{K} \left( \sum_{i=1}^{K} C_i(p) \right)^2 \right)$$ (5)

where $K$ is the number of cameras that are visible to point $p$, and $C_i(p)$ is the color of the projection point on image $i$ for $p$. Equation (5) is essentially the unbiased estimation of the color variance among different projection points on different images. Note that our proposed method is not specific to a particular photo-consistency measurement, and it can adopt any recently developed complex measurement such as the one in [16].

Because the surface update operates on vertices, what we need is the distortion measurement for each vertex. Directly using the photo-consistency measure $F(v)$ as the vertex distortion is inappropriate since it does not represent the local surface distortion around vertex $v$. Thus, to better approximate the surface distortion, we should consider the photo-consistency values of all the sampling points in a small neighboring region around vertex $v$ for the vertex distortion measure.

The challenge here is how to do the sampling. Over-sampling on the surface will significantly increase the computational complexity while under-sampling will result in inaccurate distortion measurement. Considering the number of surface points that can be observed in each image is constrained by the size and the resolution of the image, a good way is to relate the surface sampling in each local region with the corresponding local image resolution.

In particular, the surface sampling consists of two components: the sampling method and the sampling resolution. There are two major sampling methods [5], [6] reported in literature. One method proposed in [6] is to sample around a small patch on the tangent plane of a vertex. However, sampling on the tangent plane of each vertex does not provide
a good approximation to high-curvature regions. An alternative method used in [5] is to choose sample points on an orthogonal lattice of each surface triangle. But for obtaining the photo-consistency value of a vertex, only the three closest adjacent samples are used to compute the bary centric value for the vertex.

To better approximate the surface distortion, in this work, we propose to perform uniform sampling within each triangle but non-uniform sampling among different triangles through different sampling resolution. Particularly, for a triangle \((v_1, v_2, v_3)\), a set of sampling points are generated according to

\[
p = v_1(1 - m) + v_2m(1 - n) + v_3mn,
\]

where \(m = 0 : 1\) and \(n = 0 : m\). By adjusting the sample resolutions \((dn_1, dn_2)\) of \(m\) and \(n\), we can control the number of sampling points on each triangle. An example of our sampling method is illustrated in Fig. 2.

Now we need to determine the sampling resolution. Duan et al. [6] did not address this issue in their work. A method was suggested in [5], [17], which determines the sampling resolution according to the image resolution. However, this method requires back projection of each image pixel onto the 3D surface in order to find a good estimation of the sampling resolution, where back projection from a 2D space to a 3D space is a time-consuming process.

In our work, we propose a more efficient method to relate the sampling resolution with the image resolution. Particularly, for each triangle on the surface, we forward-project the triangle onto all visible image planes. We calculate the area \(A_t\) of each projected triangle and derive the sampling resolution as

\[
dn = \frac{1}{\max\{\sqrt{2 \times f(A_1, A_2, \ldots, A_n)} - 2, 1\}}.
\]

where \(A_t\) is computed as the number of pixels in each projected triangle, and \(f(A_1, A_2, \ldots, A_n)\) is a function to control the relationship between the sampling resolution and the image resolution. In this work, we set \(f(A_1, A_2, \ldots, A_n) = C \times \text{average}(A_1, A_2, \ldots, A_n)\), where \(C\) is a control factor.

We are now ready to define the distortion measures. First, we define the distortion of a triangle \(T_i\) as the weighted summation of the photo-consistency values of all the sample points in the triangle, i.e.

\[
\varphi(T_i) = \sum_{p \in T_i} \omega_p F(p)
\]

where \(\omega_p = 1/\text{valence}(p)\), \(1/2, 1\) for \(p\) being a vertex, a sampling point on an edge, and an inner sampling point, respectively. The weight \(\omega_p\) is to take into account the repeated counting of the photo-consistency values of the same boundary samples in multiple triangles. Then, the distortion for a vertex is defined as the average distortion of all the samples in the one-ring neighboring triangles, i.e.

\[
G(v) = \frac{1}{|N_s(v)|} \sum_{T_i \in N_t(v)} \varphi(T_i)
\]

where \(N_t(v)\) is the set of one-ring neighboring triangles, \(|N_s(v)|\) is the number of neighboring sampling points in \(N_t(v)\). The overall surface distortion is measured by the average per-sample distortion:

\[
\Psi = \frac{1}{|N_s|} \sum \varphi(T_i).
\]

where \(|N_s|\) is the total number of samples. Using the per-sample distortion makes the distortion measurement independent of the number of samples in the cases of dense samples.

C. Computing Distortion Gradient

Another question we need to answer is how to compute the distortion gradient along the normal \((\nabla G, N)\) in (3). In this work, we use the central difference to compute this distortion gradient as [5]. Specifically, \((\nabla G, N)\) is computed as the difference between the distortion values of the current vertex when it moves forward and backward along the normal direction:

\[
(\nabla G, N) = \frac{G(v + \eta \Gamma_{\min} N) - G(v - \eta \Gamma_{\min} N)}{2\eta \Gamma_{\min}}
\]

where \(G(\ )\) is the distortion function, \(\Gamma_{\min}\) is the minimum edge length of the surface and \(\eta \in (0, 1]\) is a control factor. In practice, \(\eta\) is chosen to be small enough so that the small displacement of the current vertex does not affect the visibility and normal vectors of other vertices. Note that when calculating \(G(v \pm \eta \Gamma_{\min} N)\), we need to re-sample one-ring neighborhood triangles around the current vertex.

D. Mesh Regularization

Although the surface flow method moves each vertex closer to the true surface, it degrades the regularity of the mesh. To ensure numerical stability, we need to maintain the mesh regularity in terms of vertex distribution and edge length ration (the ratio between the maximal and the minimum edge lengths). In our implementation, we use the similar tangential Laplacian operator as that in [18] to regularize the mesh:

\[
\frac{\partial \mathbf{p}}{\partial t} = \left(\frac{1}{k} \sum_{i=1}^{k} \mathbf{v}_i - \mathbf{p}\right) - \left(\frac{1}{k} \sum_{i=1}^{k} \mathbf{v}_i - \mathbf{p}\right) \cdot \mathbf{N}
\]
where \( v_i \) is a neighboring vertex of the current vertex \( p \), and \( k \) is the number of neighboring vertices. The time step \( \partial t \) is chosen to ensure the displacement of the current vertex to be bounded by the minimum edge length.

### E. Implementation of Mesh Optimization

In summary, the PDE-based mesh optimization step contains the following three main sub-steps: (1) determine the visibility of vertices using the z-buffering technique [19] as that in OpenGL; (2) update vertex positions using Eq. (4); (3) regularize vertices using Eq. (12). The process repeats until no further reduction is observed on the overall surface distortion.

As aforementioned, in order to have a high quality 3D surface reconstruction, the sampling resolution of each triangle needs to approximate to image resolution. This requires a dense sampling on the surface, which is the bottleneck to the processing speed of the system. Specifically, when we search for the optimal position for each vertex, we have to iteratively compute the distortion and the distortion gradient for each vertex, both of which require sampling one-ring neighborhood triangles around each vertex.

To improve the processing speed, we make use of GPU to enable the parallel processing capability for our system. In particular, considering that computing the photo-consistency value of each sample in a triangle is independent of other samples, we define an array of all the sample points in one triangle and send it to the GPU for parallel processing. We can also compute the distortions of all the one-ring neighborhood triangles of one vertex in parallel. In this way, the processing speed of our system becomes independent of the sampling resolution of each triangle.

### V. Progressive Mesh Refinement via Local Adaptive Subdivision

After applying the mesh optimization to the initial coarse mesh that consists of a small number of vertices, we obtain an optimized coarse mesh. To further reduce the overall surface distortion, more vertices need to be added. Here, the refinement process comes into play.

A common way to refine the mesh is to apply global subdivision to increase the resolution of the mesh. However, such a way does not lead to an optimal solution for rate-distortion efficient 3D reconstruction. This is because a real object generally does not have a uniform detail distribution over its surface. Some regions may have more details than others. Using a global refinement method increases vertices uniformly over the entire surface. As a result, some regions with less details are relatively over-refined and the regions with more details are relatively under-refined.

Therefore, a good solution is to give higher rates to the regions with more details. In other words, more vertices should be added to the regions with rich details or large distortions. In this work, we consider each triangle as a region and keep adding new vertices to the most distorted triangle until reaching the target rate. Our proposed refinement not only achieves rate-distortion efficient 3D reconstruction at the target rate but also facilitates the generation of an optimal progressive mesh, where the 3D reconstruction is optimal at any rate between the rate for the coarse mesh and the target rate.

In particular, we first compute the distortions of all the triangles according to (8) to select the most distorted triangle. Then, we apply a local subdivision scheme similar to [20] on the selected triangle. The subdivision scheme introduces three new vertices in the middle of the three triangle edges and links them together. To maintain the manifold topology, three triangles adjacent to the current triangle are subdivided accordingly. Fig. 3 shows one example, where the triangle ABC is subdivided by adding three new vertices \( \{N, M, L\} \) and the dash lines indicate newly created edges.

[Diagram of local adaptive subdivision]

After each local subdivision, we apply the mesh optimization method introduced in Section IV to optimize the positions of the newly added vertices. Considering that the displacements of the new vertices affect their local regions, we also update neighboring vertices of the three new vertices. For example, for the case in Fig. 3, in addition to optimizing the positions of the three vertices \( \{N, M, L\} \), we also update their six neighboring vertices \( \{A, B, C, D, E, F\} \). The updating stops when the distortion of any of the nine vertices cannot be further reduced. Note that besides updating the neighboring vertices we could also update all the vertices, but it would significantly increase the computational complexity. We would also like to point out that, before the mesh optimization, an edge swapping technique [21] is used to mitigate the mesh irregularity caused by the local subdivision. This is to ensure the numerical stability of the PDE-based mesh optimization.

### VI. Experiment Results

We carry out several experiments to evaluate the performance of our proposed 3D reconstruction system. We test on three models: Venus, Pot and Bowl. Each model is captured with 22 cameras uniformly placed in a circle where the model is at the center. The 22 captured color images with a resolution of 900×900 are segmented and calibrated in advance, and then used as the input to our proposed 3D reconstruction system.

Fig. 4(a)-(c) shows the visual results of the reconstructed Venus models at different stages in our system. It can be seen from Fig. 4(b) that the optimized coarse mesh obtained by the
developed PDE-based mesh optimization approximates well on the region with less details, however it cannot well represent the regions with details due to the rate constraint. With only 2454 vertices, Fig. 4(b) is the best we can get. Note that the result of Fig. 4(b) is the one obtained after many optimization iterations. With more vertices being added in, the regions with more details are recovered greatly as shown in Fig. 4(c) and (e). The produced mesh in Fig. 4(e) looks very close to the original surface (Fig. 4(f)) in terms of geometry distortion.

We further compare our system with the PDE-based surface flow method [6]. To have a fair comparison, we adjust the setting of the remeshing process in Section III to generate around the same number of vertices as our case in Fig. 4(e) and then use the PDE-based surface flow method to perform the vertex optimization, which produces a result in Fig. 4(d). In other words, our method consists of three steps and starts from a coarse mesh while the other method only uses the first two steps of our system and reaches the target rate at the beginning. Comparing Fig. 4(d) and (e), we can see that the output of the surface flow method [6] does not capture those regions with more details such as the eyes, the hair, the nose and the mouth of Venus as well as that using our system. Our reconstructed model in Fig. 4(c) has similar visual quality as Fig. 4(d) but with less number of vertices. The visual comparison results for another two models are shown in Fig. 5, where similar observations can be made.

In addition to showing the visual quality improvement at one rate, we generate the rate-distortion curve for each model and compare the rate-distortion curves of different methods. Fig. 6 shows the obtained curves. To obtain the rate-distortion curves for the surface flow method [6], we generate a set of initial meshes with different number of vertices, then optimize them and record their distortions at the stable state.

Fig. 6 clearly demonstrates the superior performance of our proposed system. Compared with the surface flow method [6], our system achieves lower per-sample distortion at not only one rate but also all the rates except the starting rate. For example, our system reduces the per-sample distortion about 70.5% for Venus at around 4204 vertices, 76.8% for Pot at around 9403 vertices, 47.5% for Bowl at around 8365 vertices. The reason that our system significantly outperforms the surface flow method lies in the proposed rate-distortion optimized local adaptive refinement that facilitates nonuniform vertex distribution among different regions. In other words, our system allocates more vertices to the regions with more details and processes it in a rate-distortion optimized way.

We would like to point out that although the per-sample distortion is independent of the sampling resolution when the number of samples is large, it does matter in the cases of low sampling resolution. Fig. 7 shows the average per-sample distortions of the reconstructed coarse Venus model at low sampling resolution under different optimization iterations. We study the case of applying the PDE-based mesh optimization on the initial coarse mesh and compare the results with uniform sampling (the same number of samples per triangle) and the proposed adaptive sampling described in Section IV-B. For a fair comparison, we adjust the numbers of sample points for uniform sampling and the adaptive sampling to be roughly the same.

It can be seen from Fig. 7 that the proposed adaptive sampling method achieves better distortion curve at different iterations. This is mainly because the proposed adaptive sampling facilitates non-uniform sampling that allocates more samples to the triangles corresponding to more image pixels and thus it gives a better approximation to the surface distortion. One might argue that if using high sampling resolution, the inaccurate surface distortion approximation is not a problem any more. However, high sampling resolution comes at the cost of high computation complexity. With the proposed adaptive sampling, we can achieve the same level of surface distortion approximation but with less computation.

VII. CONCLUSIONS

In this paper, we have presented a novel rate-distortion efficient multi-view reconstruction system, which consists of two major components: the PDE-based mesh optimization and the local subdivision based mesh refinement. Our proposed adaptive sampling facilitates the non-uniform sample distribution so that the surface distortion can be efficiently and accurately approximated through discrete points. Moreover, our proposed local adaptive refinement facilitates the non-uniform vertex distribution so as to allocate more vertices to the regions with more details. The extensive results show that our system significantly outperforms the original surface flow method at all the reconstruction rates.

Our system also has some limitations. First, due to high complexity, our system takes a long time (about half an hour) to produce a fine 3D model. It is highly desired to speed up the process. Second, the PDE-based mesh optimization is very sensitive. There are quite a few parameters that need to be
carefully chosen. For example, if the time step $\Delta t$ in (4) is too small, the surface evolution becomes very slow, while it becomes unstable if the time step is set to a large value. All these issues need to be further investigated.

ACKNOWLEDGMENT

This work is supported by A*STAR SERC TSRP Grant (NO. 062 130 0059).

REFERENCES


Fig. 5. Visual comparison of the reconstructed Pot (top row) and Bowl (bottom row) models: (a) optimized coarse meshes with 5308 vertices and 10612 triangles for Pot and 5296 vertices and 10588 triangles for Bowl, (b) optimized fine meshes using [6] with 29886 vertices for Pot and 10631 vertices for Bowl, (c) results of our proposed system with 29884 vertices for Pot and 10639 vertices for Bowl, (d) two input images for Pot and Bowl respectively.

Fig. 6. The average per-sample distortions of the reconstructed Venus models at different number of vertices.


