Clustering and Link Propagation for Surface Reconstruction

Dennis Maier*, Jürgen Hesser, Reinhard Männer
Chair of Computer Science V, University of Mannheim, Germany

1 Introduction

Our goal is the fast automatic reconstruction of a surface from an image sequence. We use shape-from-motion algorithms to determine position and orientation of the cameras that took the images. An efficient fixed window size correlation based dense stereo algorithm is used to compute a 3d point cloud for every consecutive image pair. These must be merged into a consistent surface model. Due to the simple but fast stereo algorithm we use the 3d data are very noisy and include many outliers. These wrong points are typically not isolated but occur clustered in regions where the correspondence search of the stereo algorithm matched to wrong pixels possibly due to repetitive texture in the images. Furthermore the fixed window size will result in an over estimation of the surface near object boundaries. A single reconstruction may contain holes, however other views might reconstruct the surface at this location.

The basic idea to solve the problem is to use local information obtained from the individual reconstructions and to propagate this information into clusters of the 3d data points that are computed by Euclidean proximity and normal similarity. The proposed method requires only a small amount of memory, is time efficient and does not require any preprocessing of the data as opposed to many other existing techniques.

A standard technique to geometric fusion of 3d point clouds is a volumetric approach which requires a lot of memory and is much slower than our proposed method. Existing techniques like polygon mesh zippering, reconstruction from unorganized point sets, ball-pivoting and voronoi-based triangulations, first generate meshes consisting of a large number of triangles and then decimate the number of triangles using mesh simplification algorithms.

2 Clustering and Link Propagation

For every individual reconstruction we generate a triangulation which is used to compute an estimated surface normal for every 3d point. The points of the individual reconstructions are easily meshed by triangulating the originating pixels of the stereo images.

Using an appropriate data structure to facilitate fast search for neighboring vertices, we build clusters of nearby 3d points. Starting with an arbitrary point, we merge the closest point into this cluster if the angle between their normals is below a certain threshold. The position of the cluster is updated in every step to the center of mass of all points contained in the cluster and the cluster normal is set to the average of the normals of all contained points. We grow the cluster until its size reaches a predefined cluster size or the next closest point cannot be merged because the angle between their normals is greater than the threshold. This way clusters are large in areas of low surface curvature and small in areas of high curvature. When a 3d point is merged into a cluster we store the ID of that cluster in the point so the links from the triangulations of the individual reconstructions can be efficiently propagated into the clusters after the clustering stage. Clusters that contain only reconstructed points from the same view are discarded. If no other reconstruction contains points close enough to be clustered or if their normals differ by more than the allowed threshold, these points are considered unreliable and should thus be removed from the reconstructed surface.

For every cluster we iterate over all points contained in it and add a link to all clusters the points are linked to in the individual triangulations, if such a link does not already exist and does not point to itself. Then we first determine all possible triangles that can be generated with the links that were propagated from the individual reconstructions of the previous step. For every pair of outgoing links from a cluster we check whether the two clusters to which the links point to are themselves linked in which case a triangle is generated.

This may lead to redundant or conflicting triangles for many clusters if a lot of individual reconstructions are merged. The decision which triangles to keep are made locally. We first check for redundant triangles (i.e. two triangles that contain the same four vertices as two other triangles) and keep those triangles with the smallest total sum of edge length. We then look for closed triangle fans and if more than one possible solution is found we keep the one with the smallest area to make sure the cluster is linked with those clusters that are closest on the surface.

Since we want to reconstruct manifold surfaces, we finally check that each edge is contained in at most two triangles. If more than two triangles share the same edge, we delete the triangle whose normal differs most from the normals of the clusters that span the edge until only two triangles remain. This local consistency implies global consistency.

3 Conclusion

The proposed method for surface reconstruction is very fast. It takes about 3 minutes to compute the point clouds for 36 images of a turntable sequence with a fixed window size, correlation based stereo algorithm on a 800Mhz PentiumIII with 512MB RAM. The clustering of the resulting 1.7 million points takes about 20 seconds and building the triangle mesh less than 1 second.

Instead of first creating a high resolution mesh that is reduced afterwards, we cluster the 3d points in the first step, propagate the link information of the individual reconstructions, and decide locally which resulting triangles best approximate the surface. Thereby we directly determine the vertices of the final surface mesh and use the propagated link information of the individual reconstructions to create the triangulation. A simple way to determine which image(s) to use for texture generation is to check which view(s) is contained in all three cluster vertices of a triangle.

*e-mail: dennis.maier@ti.uni-mannheim.de