Low-Cost Acquisition of 3D Interior Models for Online Browsing

Filip Mikle, Matej Minárik, Juraj Slavíček, Martin Tamajka*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
imaginecup2014@googlegroups.com

Building a dense 3D model in a usual 3D editing tool requires a lot of time and a significant skill in this field. Using our solution, everyone, even non-technically skilled person is able to model real places in a few minutes. Modeling interiors this way looks much more like filming than 3D modeling. It takes only a few more minutes to get a 3D model than to take a set of pictures of interior.

Naturally, there've always been efforts to make this process automatic. Laser scanners provide professional, accurate, but expensive way of modeling of interiors, which generally requires a lot of time and preparation. In comparison to laser scanners, our solution requires affordable and commonly used device - Kinect for Windows. It's also much faster to create 3D model using our product.

The first part is a scanning application, designed for real estate agents. The second part is a browsing web application, designed for ordinary people. Customers of real estate agency can easily walk through different interiors previously scanned. There is also information about our project and information for real estate agents. When the scanning is finished, scanning application processes the data and generates group of object files. Subsequently, agent uploads these files through our web application and the model is ready to be browsed.

We decided not to develop from absolute scratch. Scanning and pointcloud-to-mesh transformation and optimization parts are mainly based on features provided by open-source library PointCloudLibrary (PCL) and its dependent libraries. The Web-application is a part of our product and it's based on ASP.NET.

We found out, that we need to remove redundant information from our 3D models. This redundancy, arising in scanning-phase, causes enormous increase of model's size (every point in space could be contained in model multiple times). That's why we developed *Voxel Tree*. It allows dynamic addition of environment features to model, guaranteeing, that every information will be in model contained only once. It

* Supervisor: Jakub Šimko, Institute of Informatics and Software Engineering

Spring 2014 PeWe Workshop, March 21, 2014, pp. 57–58.

also allows to choose density of model (the size of atomic part of model - voxel, in meters).

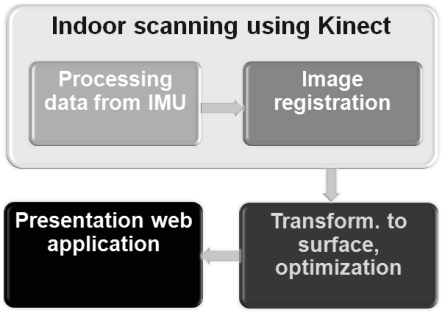

Image 1 - Architecture overview

The alignment of neighbor frames is reached using the computer vision techniques. One of algorithms used for this purpose, is Iterative Closest Point (ICP) [1]. This algorithm would be costly and not affordable in real time, if it would work with all the points got from Kinect. This is, where keypoints come to use. We extract keypoints based on depth and also on color information from environment, ignoring those one, that do not have known distance from camera (unknown depth). Everywhere, where appropriate, we use parallelism, recording significant increase on performance. Although the contribution of parallelism on CPU is large, our future plans lead to make as many as possible (and appropriate) computations on GPU (currently, we process to 3 frames per second).

One of the inputs to Iterative Closest Point algorithm is initial estimation of device´s orientation. This additional (and very important) information is taken from IMU (inertial measurement unit), electronic device that measures and reports on a craft's velocity, orientation, and gravitational forces, using a combination of accelerometers and gyroscope and magnetometer. Obtained measurements are converted to Euler angles, which represent current orientation of the Kinect.

Point clouds obtained in previous parts of scanning process are not the best for further manipulation, so we represent our models as a surface. We use a method that creates bounding box which is a 3D object containing all points of original cloud. Bounding box is then divided into uniform voxel grid which is essentially a little cube. Each point in the original point cloud is then assigned to the corresponding voxel according to its 3D coordinate. Then we make each wall of each voxel solid or transparent taking into account if wall can be seen from concrete position of Kinect.

This abstract is based on paper accepted to the Student Research Conference (IIT.SRC 2014).

This work was partially supported by the Institute of Informatics and Software Engineering, Slovak University of Technology in Bratislava.

References

[1] Besl, Paul J. and McKay, Neil D. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* February 1992, Vol. 14, 2.