Geospatial Management and Utilization of Large-Scale Urban Visual Reconstructions

Clemens Arth, Jonathan Ventura, Dieter Schmalstieg
Institute for Computer Graphics and Vision,
Graz University of Technology
Email: {arth,ventura,schmalstieg}@icg.tugraz.at

***Abstract*—In this work we describe our approach to efficiently create, handle and organize large-scale Structure-from-Motion reconstructions of urban environments. For acquiring vast amounts of data, we use a *Point Grey Ladybug 3* omnidirectional camera and a custom backpack system with a differential GPS sensor. Sparse point cloud reconstructions are generated and aligned with respect to the world in an offline process. Finally, all the data is stored in a geospatial database. We incorporate additional data from multiple crowd-sourced databases, such as maps from *OpenStreetMap* or images from *Flickr* or *Instagram*. We discuss how our system could be used in potential application scenarios from the area of Augmented Reality.**

I. Introduction

Structure-from-motion (SfM) algorithms for building large-scale reconstructions of urban environments have attracted significant attention recently. Interesting application scenarios arise from the newly generated and maintained data. Tourists may plan their trip in advance by traveling to the holiday destination in virtual space. Augmented Reality (AR) browsers can accurately register the current view in a global context and provide a significantly higher level of performance (Figure 1). Any application in pedestrian navigation, advertising, social networking or gaming using mobile gadgets is boosted by the availability of densely sampled data from the environment, which can be processed and distributed through the cloud.

Despite great advances in the area of reconstruction itself, methods and procedures to efficiently *acquire* and *organize* the vast amount of diverse data types are still missing or insufficiently documented. Usually large companies such as Microsoft or Google keep this know-how as trade secret. Teams of experts in individual areas like data acquisition, processing and organization are employed to track down individual subproblems in a large project, such as in Google's *Street View* for example. Almost unlimited financial resources for hardware acquisition relieve the burden to take on highly complex computational processes. In a small or medium-size company, however, it is almost impossible to employ experts in each domain. Furthermore the budget for hardware is clearly restricted, implicitly limiting the scope of action.

In this work we describe our system to efficiently generate and maintain large-scale point-cloud reconstructions of urban environments. Third-party data, such as maps from *OpenStreetMap* or images from *Flickr* are integrated to bring global context to our self-generated data. We stress the fact that hardware costs for our setup are low compared to professional setups [1]. All algorithms and tools are available as open-source or can be implemented even by a motivated graduate student in a reasonable amount of time. The output of our approach does not achieve the precision and completeness of more expensive setups, however, we argue that for a small or medium-size company it is reasonable to maintain such a system and to set up a solid business model, in the area of advertising or AR for example.

Fig. 1. *Top*: Layar augmented reality browser using low-accuracy GPS and compass for positioning; *Bottom*: concept image for application of our work: multiple geographic content sources are mixed in a high-accuracy augmented reality view.

The rest of the paper is structured as follows. In Section II we describe our data acquisition setup in detail. Section III covers our reconstruction algorithm. In Section IV we outline our database setup and show how it can be used to efficiently organize and query in a geo-spatial context. Section V is devoted to an application scenario in AR. The paper concludes with final remarks in Section VI.

Fig. 2. Left: data acquisition system, consisting of the backpack, the *Point Grey Ladybug 3* camera mounted on a tripod and the antenna of the *Novatel* GPS receiver on top of the camera. Right: a user operating the acquisition system.

II. Data Acquisition Hardware

The most important part of our system is the acquisition setup. Since data acquisition is a tedious and time-intensive task, the goal is to acquire as much information about the environment as possible while visiting a single location only once. Our design uses a omnidirectional camera mounted on a backpack for maximum mobility. The camera rig is a *Point Grey Ladybug 3* spherical camera system consisting of six individual 2MP cameras[1]. The system features a global shutter and allows streaming of JPEG compressed images at 15 FPS through a Firewire S800 interface. The camera is mounted on a carbon-fibre tripod fixed to the aluminum frame of a hiking backpack.The backpack and a user carrying the setup during a data acquisition session are shown in Figure 2.

Because we want to globally register the data with the highest accuracy possible, we use a *Novatel OEMv2* RTK Differential GPS receiver for acquiring GPS estimates[2]. Since we do not use the sixth camera of the Ladybug rig which points upwards, the antenna is mounted on top of the rig giving the best possible satellite reception. The GPS receiver automatically incorporates corrections from a base station. To receive these correction data, a connection to the server of the service provider is required. We use an *Apple iPhone 4s* mobile phone as a wireless hotspot. Since the size of the data packets is very small and updates are usually provided only once a second, a standard UMTS/HSDPA connection is sufficient. Using the correction service, the accuracy of the GPS estimates can achieve about $5cm$, which is considerably higher than the estimates provided by off-the-shelf receivers used for hiking or those built into standard smartphones. The receiver is powered by a separate battery pack.

For recording the large amount of image data we use a laptop. Because there are only a few laptops available on the market featuring a Firewire S800 interface, we chose an *Apple MacBook Pro* (early-2010) with a 2.4 GHz Dual-Core CPU and 8 GB of RAM. For image recording we use the standard software bundles provided by *Point Grey* and *Novatel*. Because this software is only supported on Windows, the mobile PC runs Windows 7 64-bit. The recording software for the image material automatically annotates each frame with the corresponding GPS estimate from the receiver. The mobile PC is stored in the backpack during data acquisition, so for storing the data a 250GB SSD drive is used. Since the mobile PC powers the camera rig, the standard battery keeps the system operational for about 1 hour. For longer recording sessions, a battery extension pack and a second SSD drive could be added.

We clone the desktop on a small *Mimo* touch screen which is attached via USB[3]. The user can control the entire setup conveniently using this interface, while the rest of the equipment is stored in the backpack. The mobile PC and it's interfaces are depicted in Figure 3.

The entire cost of our current system is about €31,200, as evaluated in Table I. Depending on the actual components used and market conditions, the total amount might vary, however, we argue that this is a reasonable amount of money to spend for such a setup, compared to professional setups such as those employed by Google [1].

Fig. 3. Hardware setup in detail. *Apple iPhone 4s*, *Mimo Touch 2*, Battery Pack and *Novatel* GPS receiver, connected via USB hub to an *Apple MacBook Pro*.

Component	Avg. Price
Mimo Touch 2 Touch Screen	€200
Tatonka Backpack and *Manfrotto* Tripod	€400
Apple iPhone 4s Mobile Phone	€600
Apple MacBook Pro Laptop	€2,500
Point Grey Ladybug 3 Camera Rig	€12,500
Novatel OEMv2 GPS Receiver	€15,000
Total	**€31,200**

TABLE I. Approximated hardware cost for each individual component.

III. Urban Reconstruction

In the following we describe our framework to reconstruct large areas using the hardware setup described previously.

[1]Point Grey Ladybug 3: http://www.ptgrey.com/products/ladybug3/ladybug3_360_video_camera.asp

[2]Novatel OEMv2: http://www.novatel.com/products/gnss-receivers/oem-receiver-boards/oemv-receivers/oemv-2/

[3]Mimo Touch 2: http://www.mimomonitors.com/products/mimo-touch2

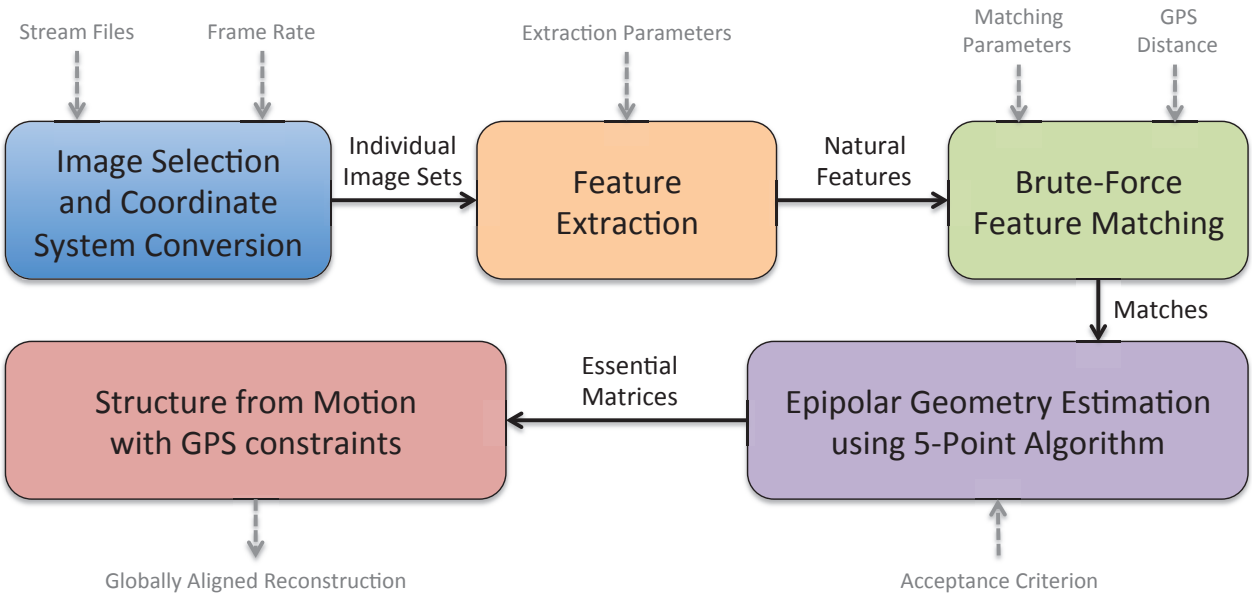

Fig. 4. Flow chart of our Structure-from-Motion reconstruction system. After frame selection, natural features are extracted for each frame and those features of locally neighboring frames are matched. The matches are fed into a geometry estimation module, which delivers essential matrices to the final SfM module. As a result we obtain fully globally registered sparse point-cloud reconstructions.

Discussing all related approaches in detail is out of the scope of this paper; however, the interested reader is referred to a survey about the state-of-the-art in urban reconstruction by Musialski et al. [2].

The image acquisition software records the images for the individual cameras of the rig annotated with the GPS estimates into a set of files. These files are transferred to a desktop PC prior to processing. The image stream is recorded at 15 FPS (here a frame refers to a set of six images from the six individual cameras). However, we do not use every individual frame for reconstruction since the baseline change – and therefore the change of the image content – is too small between individual consecutive frames. We subsample to 1 FPS, selecting only one frame each second and discarding all others. This is roughly equivalent to having a single snapshot of the environment for every meter of track to be reconstructed, based on normal walking speed with the backpack.

Several steps of pre-processing are applied to prepare the images for the structure from motion pipeline. Radial lens distortion is removed from each image according to a one-time calibration step. Due to the global shutter, some images might be under- or overexposed. We have found this to have negative impact on the subsequent feature extraction and matching stage. We therefore apply a non-linear tone-mapping to each image to improve the overall contrast and brightness of the image. Since longitude and latitude do not relate to a Euclidean coordinate system, we convert the GPS coordinates for each snapshot into UTM coordinates and pass them along with estimated altitude for each image.

As a next step for each individual image we extract invariant features. We use SIFT [3] since it is a well-known state-of-the-art algorithm and use the implementation from OpenCV[4]. We do not tune the extraction parameters but rather use the standard settings.

After extraction we match the features between images from individual frames in a brute-force manner, using the standard best-vs-second-best ratio matching criterion [3]. We also specify a geographic distance parameter, such that the images of a single frame are only matched against frames within a certain radius. Because we are working with UTM coordinates, we can specify this parameter in meters and usually use a value between 5 and 10. Thanks to the GPS measurement, some loop closures are easily found using this method, without resorting to costly exhaustive matching of all frames.

The feature matches between frames are passed on to a modified five-point algorithm [4]. We treat the five individual pinhole cameras as a single omnidirectional camera. (The rotation between cameras is determined in the calibration step.) The geometry of the essential matrix then applies to omnidirectional image just as it does with single pinhole camera images. However, care needs to be taken when using this representation. First, from the intrinsic calibration of the camera rig we discard the baseline of the individual cameras and keep the rotational parts only. In other words, we treat all images of the frame as if the cameras had a common center of projection. For the five-point algorithm, all feature points from the individual images are therefore converted to rays emanating from a common center. As a second step, we remove the check for observations that fall *behind* the camera center, as *behind* has no reasonable meaning when using a omnidirectional camera. Finally, in determining the support for a hypothesis, we discard all matches which are viewed under a very narrow angle. This step is required to remove the errors introduced by matches that are only in the forward (or backward) direction of movement. These matches give only weak constraints on the translational part of the actual hypothesis. The algorithm is run in a robust RANSAC scheme

[4]OpenCV: http://opencv.willowgarage.com

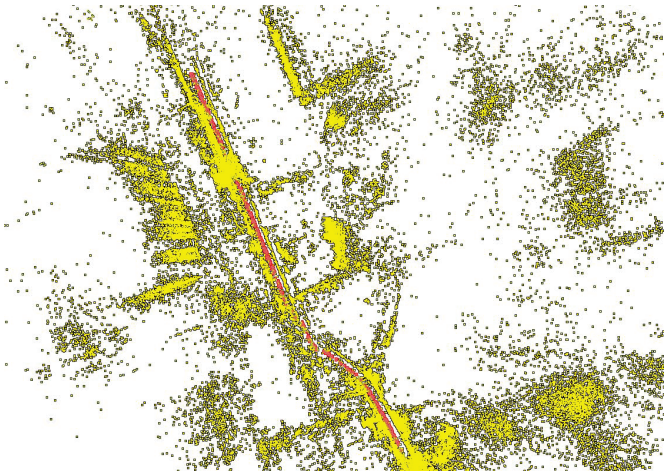

Fig. 5. Top-view of a sparse point-cloud reconstruction. The frame locations are depicted in red, while the points are depicted in yellow.

and usually finds the right solution within a few iterations. Note that we do not expect much error to be introduced by ignoring the translational part in the intrinsic calibration of the camera rig, since that translation is negligible compared to the distance to the objects visible in the images.

The estimate of the geometry between individual frames is passed to the SfM module. Frames are incrementally linked together starting with the location which has the most neighboring frames within the previously defined distance radius. The initial scale of the reconstruction is estimated using the UTM coordinates for the individual frames. In doing so we get the scale of the reconstruction immediately from the first pair of frames. For each subsequent frame, we estimate the camera pose and link the new frame to the existing reconstruction. This procedure is repeated until all frames are linked into a single reconstruction. For every ten frames inserted into the reconstruction we run a bundle adjustment (BA) step. In BA, we minimize the reprojection error of the triangulated 3D points by adjusting the camera poses and point locations. As an additional constraint, we minimize the distance between the frame position estimates and the UTM coordinates that we acquired with the GPS receiver. As a result we obtain a fully registered sparse point-cloud reconstruction, as shown in Figure 5.

The reconstruction process highly benefits from the compact representation of the multi-camera head as a single rig. In urban environments, repetitive structure on buildings is prevalent, which results in a large amount of false matches between individual images. However, since the modified 5-point algorithm takes all matches from the entire rig into account simultaneously, achieving a wrong result for the epipolar geometry between consecutive acquisition moments is almost impossible. Mismatches in the preceding stage are therefore removed automatically. For a typical outdoor reconstruction as described in Table II, the final mean reprojection error is 1.49 pixels. The accuracy of the global alignment heavily depends on the quality of GPS reception, which can be unfavorable in narrow alleyways or close to high buildings. The largest deviation is naturally to be found at the borders of the reconstruction. In the typical case described, the maximum error is $\leq 2^\circ$ in rotation and $\leq 50cm$ in translation.

Fig. 6. Exemplary view of a reconstruction of the *Grazer Hauptplatz* using *PMVS* [6]. Since the models are registered, the point cloud accurately aligns with the polygonal building models visualized in *light grey* color.

Outdoor reconstruction	
Number of panoramic images	193
Number of single pinhole camera images	965
Single Image Resolution	2504x1628 pixels
Number of 3D points	371,226
Number of 2D features	3,979,585
Approximate area	6,742 m^2

TABLE II. INFORMATION ABOUT A TYPICAL OUTDOOR RECONSTRUCTION (IN THIS CASE THE AREA AROUND *Grazer Hauptplatz*).

Our procedure for reconstruction is outlined in Figure 4. Note that we use our own implementations of the individual modules. However, there are open-source frameworks available which can be modified accordingly, such as *Bundler* [5] for example. As an optional step, the sparse point clouds may be pseudo-densified by using *PMVS* [6] to generate more visually pleasing reconstructions. An exemplary view is depicted in Figure 6.

An important question arising is the amount of data acquired over time and the required amount of time for reconstruction. For a single minute of recording using the camera rig, approximately 3.3 GB of image data is recorded. Assuming an average walking speed of a person with about 4 km/h, this amounts to about 5 GB of data per 100 meter walking distance. Reconstructing a road section of 100 meter length takes about 8 hours on a desktop computer with Intel i5 2.5GHz quad-core CPU and 8 GB of RAM. However, our pipeline is not fully optimized and this performance could be improved.

IV. GEOSPATIAL DATA MANAGEMENT

Because our urban reconstruction output is registered to a geographic coordinate system, we propose the use of a geospatial database to store it. Such a database allows for simple and complex geographic queries, as well as mixing of different data types, provided that all data has an associated geographic extent. In our implementation we choose to use *PostgreSQL* since with spatial query support provided by the *PostGIS* extension. Easy visualization of the geospatial data is provided by *Quantum GIS*[5] which directly interfaces with

[5]Quantum GIS: http://www.qgis.org/

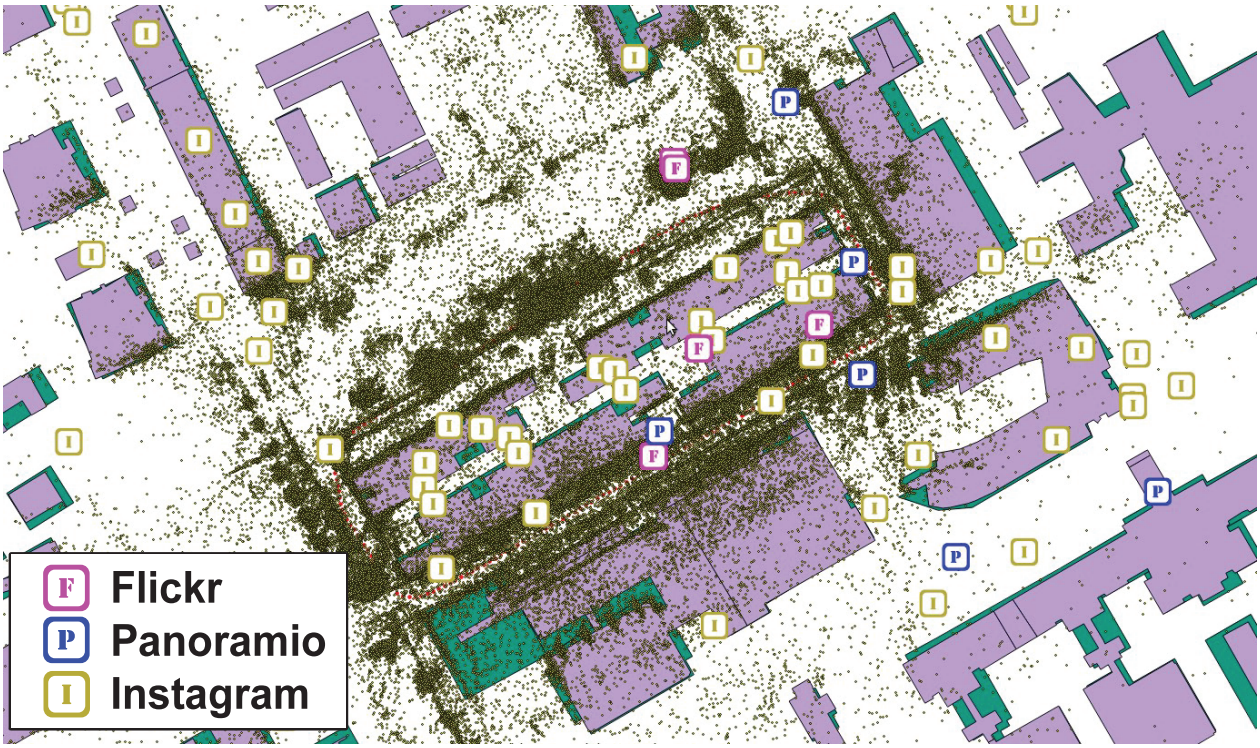

Fig. 7. Snapshot from QGIS. The camera positions as marked as red circles and the buildings from OpenStreetMap and BEV as blue and green polygons, respectively. Overlaid are the locations of Flickr, Panoramio and Instagram images as icons.

the database and renders 2D overhead views of points and polygons.

The structure-from-motion reconstruction is organized by creating tables for reconstructions, images, features and 3D points. After processing each sequence, we import all 3D points, camera locations and image features including descriptor vectors into the database.

Using the geospatial database, it is possible to store other geographic data alongside the structure-from-motion reconstruction. One useful resource is *OpenStreetMap* which provides building extents, road maps, and other labeled areas of interest. A second data source for mapping is kindly provided to us by the local land surveying office (in Austria, this is called the Bundesamt für Eich- und Vermessungswesen or BEV). This data contains building outlines and also information about the actual height of buildings which was determined by aerial laser scans.

It is also possible to import geo-referenced social media content such as publicly shared photographs into the database. We import images from *Flickr*, *Instagram* and *Panoramio*. Social network sites such as *Facebook* allow for geo-tagging of status updates and other posts, which could also be queried and introduced into the database.

A snapshot of all the data imported for the area around our campus in Graz is shown in Figure 7. From the image it is clearly visible that the data sources do not share the same level of accuracy.

Our database can now be queried efficiently for certain information the user is interested in. An exemplary query is the search for the closest buildings in a certain vicinity:

```
select * from buildings where ST_DWithin(
  center, ST_GeometryFromText(
  'POINT(<lat>,<lon>)',4326), <radius>);
```

In this case we have to provide a GPS position and a search radius as parameters to the query. For querying the URL and other metadata of *Flickr* images which were taken past January 1 2010, one can use the following query:

```
select * from flickr.images where ST_DWithin(
  position,ST_GeometryFromText(
  "POINT(<lat>,<lon>)",4326), <radius>) AND time >
  "2010-01-01 00:00:00";
```

The use of the database is very simple and flexible, at the same time offering a large number of possibilities for providing information in an application scenario.

V. Augmented Reality Application

The unified geospatial database allows for informational displays of content from various geographic sources in a combined visualization. With the addition of geo-registered imagery and a 3D point cloud model into the database, we can produce visually annotated photographs and live videos, i.e. augmented reality mashups.

To create an augmented reality "mashup" using these various geographic data sources, we need to determine the camera pose of the mobile phone. Whereas current commercially available apps use GPS, compass and inertial sensors for coarse position sensing, much higher accuracy can be achieved by using computer vision techniques to exactly match the camera image to the 3D visual model stored in the database.

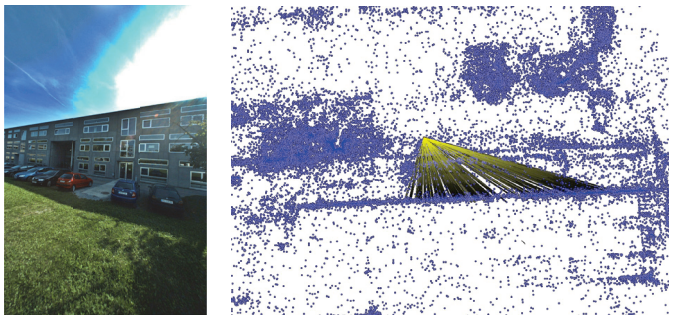

Fig. 8. Result for querying the 6-DOF pose of an image, depicted on the left, within our reconstruction. On the right, the point cloud of the reconstruction is depicted. The point correspondences that were used for estimating the pose are depicted as yellow lines.

Recent research has approached the topic of highly accurate visual self-localization in outdoor environments. The camera pose can be estimated on the mobile device itself by distributing the environment model to the mobile phone [7], [8], [9]. Alternatively, a cloud-based service can be employed to provide device localization [10]. Figure 8 shows a top-down view of the result for localizing a single image within our reconstruction.

Figure 1 illustrates an AR application concept where multiple geographic content sources are mixed in the augmented camera view. The unified geospatial database allows for querying relevant augmentations from various domains: 3D building outlines, labeled points and areas of interest, street names and maps, and social media content such as geo-located images and status updates.

Beyond passive displays, another compelling AR application is enabling the addition of geospatial content by end-users. For example, in our concept image (Figure 1) an office window has been outlined on the building wall, the extent of which was provided by GIS data. A personal text message was then attached to the window object. This kind of functionality goes beyond current social media apps which allow for simply tagging a photograph or status update with a GPS reading. Using a mashup of geospatial data sources in the AR camera view as described above, users can highlight objects, add geo-referenced augmentations to them, and share their content publicly or within their social network.

In a commercial context, one can imagine innovative advertising campaigns which show engaging 3D content situated in a commercial space. Cultural heritage sites could use the geospatial AR framework to visualize the historical appearance of a ruins, or to annotate points of interest with additional media. Industrial companies could use this technology for AR construction monitoring or collaborative design. The creation and management of infrastructure to support these applications is made much easier by using a geo-referenced visual reconstruction and a geo-spatial database as we describe.

VI. Conclusion

In this paper we described our approach to efficiently acquire and maintain large-scale urban reconstructions using a cost-effective framework. We elaborately outlined our hardware setup and the software components used and explained, which algorithms are required to generate the sparse point cloud reconstructions. A description of our management system is given, including examples of geospatial queries on the underlying database. Finally we described applications of our system to the area of Augmented Reality.

We want to stress the fact that the cost for building our setup is reasonable for a small or medium size company. In our opinion, it allows the proposal of a business case for an application in the area of AR or in another mobile context, such as tourist guidance or advertising. Such a system can immediately be used to improve existing applications, such as AR browsers to a level of performance far beyond what is currently available.

It is obvious that the quality and extent of the reconstructions we obtain cannot directly compete with professional setups used by large companies; however, we argue that depending on the actual use case the proposed framework is suitable for creating sufficiently accurate and dense information in a limited area for building interesting custom applications and services.

Acknowledgements

This work was funded by the Christian Doppler Laboratory for Handheld AR. Thanks go to the BEV Graz which kindly and instantly provided us with data from aerial laser scans.

References

[1] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver, "Google Street View: Capturing the World at Street Level," *Computer*, vol. 43, 2010. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5481932&tag=1

[2] P. Musialski, P. Wonka, D. Aliaga, M. Wimmer, L. van Gool, and W. Purgathofer, "A Survey of Urban Reconstruction," in *EUROGRAPHICS (State-of-the-Art-Report)*, 2012.

[3] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. Journal Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[4] D. Nistér, "An Efficient Solution to the Five-Point Relative Pose Problem," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 26, no. 6, pp. 756–777, Jun. 2004.

[5] N. Snavely, S. M. Seitz, and R. Szeliski, "Modeling the World from Internet Photo Collections," *Int. Journal Computer Vision (IJCV)*, vol. 80, no. 2, pp. 189–210, Nov. 2008.

[6] Y. Furukawa and J. Ponce, "Accurate, Dense, and Robust Multiview Stereopsis," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32, no. 8, pp. 1362 –1376, aug. 2010.

[7] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg, "Wide Area Localization on Mobile Phones," in *Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2009, pp. 73–82.

[8] C. Arth, M. Klopschitz, G. Reitmayr, and D. Schmalstieg, "Real-Time Self-Localization from Panoramic Images on Mobile Devices," in *Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2011, pp. 37–46.

[9] C. Arth, A. Mulloni, and D. Schmalstieg, "Exploiting Sensors on Mobile Phones to Improve Wide-Area Localization," in *Int. Conference on Pattern Recognition (ICPR)*, 2012.

[10] J. Ventura and T. Höllerer, "Wide-area Scene Mapping for Mobile Visual Tracking," in *Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2012.