A MOBILE SYSTEM FOR MULTI-VIDEO RECORDING

Lukas Ahrenberg Ivo Ihrke Marcus Magnor
{ahrenberg, ihrke, magnor}@mpii.de
Graphics-Optics-Vision
Max-Planck-Institut für Informatik
Germany

ABSTRACT

We present a portable system to record synchronised, multi-video data for vision applications such as 3D reconstruction and video-based rendering of dynamic scenes. The aim of the project is to gain access to a greater number of scenes than what a static, wired indoor studio allows for. The portable acquisition system is constructed from a number of independent modules, each consisting of a FireWire camera and a laptop. Our software utilises wireless networking making the system behave like a tightly coupled unit without requiring the modules to be physically connected to each other. The scheme also includes an external calibration method suitable for general scenes. The system is scalable and allows for easy transportation and ad-hoc setup and configuration. We present recent results acquired in the field and their use for free-viewpoint video.

INTRODUCTION

Many of today's vision and image-based methods are heavily dependent on multiple video input data. In most cases, these methods further assume the cameras to be calibrated and the video streams to be synchronised on a frame to frame basis. To acquire such videos a multiple camera studio is used.

These studios are often rather rigid indoor installments, utilising a lot of hardware. However, several interesting types of scenes, such as wildlife, sports, and cultural events, cannot be shot in a studio and require on-location recording. While robust and powerful, a studio has the limitation of being static and immobile; If the scene does not fit inside the studio it cannot be recorded. In this paper we present the design of a portable, ad-hoc scheme for recording of out-of-studio scenes. The system is modular and based on FireWire-cameras connected to portable computers.

For calibration, we use a robust and simple method based on tracking a marker object through the scene.

PREVIOUS WORK

Several indoor, multi-video recording studios have been built. Kanade et. al. have constructed a 3D Room containing a large number of cameras mounted to the walls and ceiling [8]. Theobalt et. al. are working with a black-clothed studio consisting of 8 cameras to capture human motion [11]. A similar system has also been used by Matusik et. al. for real-time capturing and rendering of visual hulls [9]. While these systems have shown excellent results, they all have in common that the setup is constrained to a studio environment or designed for special scenes. Our setup is designed to be portable while delivering satisfactory recording results. The main difference in functionality between our system and the ones mentioned above is that we only record synchronised video, while other systems provide additional on-line processing of the data, such as background subtraction or feature matching. In our system, after recording, all data is transferred to an off-line system for further processing.

Wilburn et al. have constructed a massive multi-camera array suitable for recording a large amount of dense video-streams [14]. The array is modular and can also be reconfigured to suit different recording situations. Using a smaller amount of cameras it might be possible to build a portable system. The current implementation, however, is based on a RAID server architecture and aimed at an indoor environment.

Our calibration method does not require a calibration object that is visible to all cameras at the same time. Instead a virtual calibration object is constructed by tracking a coloured ball over time. Earlier approaches to external calibration with help of virtual calibration objects use a flashlight or LED in a dark room [1, 2]. However, the generality of our system would be severely limited if we would require that it should be possible to cover the scene in darkness.

CONCEPTUAL CONSIDERATIONS

The goal of this work is to design a recording system that can be folded up, transported and set up almost everywhere without too much effort. This requires a

compact and adaptable system, allowing for recording and calibration in any environment.

The requirement of a multi-video recording system is to provide a number of *synchronised* video streams. That is, the Nth frame in all video streams should be captured at the same instance of time, with only minor variations. This property is essential for several image-based rendering and vision applications, such as [11]. However, proper synchronisation can be hard to achieve even in tightly coupled systems. In the case of software driven triggering, as in our system, the overhead introduced by system calls, operating system scheduling and driver implementation can interfere with the overall synchronisation.

For a portable system, we would like to add three more requirements aside from the synchronisation: mobility, modularity and independent power supply.

To meet these requirements, a somewhat different system than the ones employed in a studio must be developed. In a studio, the cameras are operated by servers used for storage and, in some cases, also for triggering and post-processing of the data. Inter-server communication is performed using a standard cable-network, and both servers and cameras are powered using the electric grid. Some systems also perform synchronisation via a triggering cable connecting all cameras to a master clock. This situation is depicted in Fig. 1.

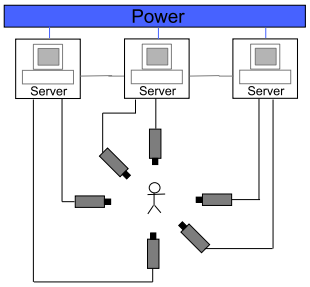

Figure 1: *Idealised studio layout. The cameras are connected to servers and sometimes also synchronised via a triggering cable (not shown in figure). Server communicate using a cable network. The power and network connections constrain the portability of the setup, the set-up time and the possible camera configurations.*

While powerful, this setup does not meet our requirements in a number of basic ways. For performance reasons, servers are typically stationary computers, which are cumbersome, hard to move and consume considerable amounts of electric power. Also, the whole system is in essence one physically inter-connected unit: cable-paths can be found between all parts of the studio. While not directly crippling the camera configuration, the wiring can constrain the general positioning of the system. Fixed length cables might directly interfere with camera placement, constraining possible positions of other cameras. Finally, server communication requires a wired network infrastructure.

To meet the requirements stated above, we designed a system built up from *camera modules*. A module is completely stand-alone, independent of any infrastructure, and is not physically connected to any other module. Each one consists of a camera, a controlling computer and a battery. To remove the cables we use wireless networking for all inter-module communication, including camera synchronisation. The concept of our system is shown in Fig. 2.

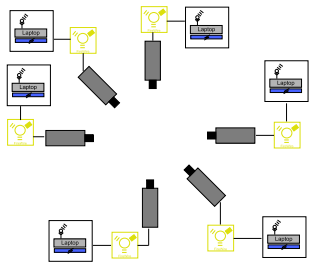

Figure 2: *Modular setup. Each module consists of a FireWire camera connected to a laptop and a battery. All inter-modular communication is performed via wireless networking. Note that this makes all modules stand-alone.*

In a studio environment, system calibration is a rare procedure: it is performed when cameras are mounted or reconfigured, or when some other radical change to the recording environment is made. In an adaptable system, however, reconfigurations are much more frequent and thus the need for flexible, easy to use calibration is greater. The calibration method is therefore required to be general and robust, performing well for a wide range of camera setups and recording environments.

IMPLEMENTATION

Hardware

Each camera module consists of the following hardware:

- Sony DFW-V500 IEEE1394 camera
- P4-1700MHz Laptop equipped with 512 MB of RAM and 802.11 wireless network.
- 12 V battery to power the camera.
- Tripod

.

All parts of our modules are off-the-shelf hardware, with the only modification being the power cable from the battery to the FireWire card-adapter. An image of the equipment can be seen in Fig. 3.

As stated above, we do not want to limit the system's usability by requiring external powering or cable-based communication.

Figure 3: *The module-components of our system. Tripod, laptop, camera, battery and inter-module cabling. The total weight is less than 10 kg. It all fits into a backpack for easy transportation.*

Resolving the issue of external power is quite easy in the case of the computer by using laptops instead of stationary PCs. However, the cameras draw 4 watts of electrical power which cannot be provided by the laptop via the FireWire port. As a solution we use a CardBus adapter model allowing for an external power source, and an additional compact 12V battery to power the camera. Although this solution adds some extra weight to the module, the battery is still rather handy.

Instead of using a cable network, we use the 802.11 wireless network. An additional feature of 802.11 networks is the ability to run in ad-hoc mode. This means that additional camera modules can be added or removed without reconfiguration of the existing setup, creating a very flexible and scalable system.

Software

Camera triggering. We have developed client-server software running on the laptops that controls most camera and recording features. A client program running on an operator laptop broadcasts commands to the servers over the wireless network, giving the user access to the whole system as one unit. The current implementation runs on a Windows 2000 system using the Carnegie Mellon 1394 FireWire drivers [13] to operate the cameras.

For synchronised recording all cameras must be triggered at the same time. Because all communication within our system is done via the wireless network, the triggering has to be performed by sending a triggering message to all modules. To keep every frame in sync during a continuous video capture two strategies can be employed: a) Sending a triggering command over the network for every frame, or b) Sending only the start pulse of the recording and then leaving it up to each module to individually keep the pace. We have chosen to implement a variant of b) in order to be less dependent on the communication channel.

It is important to find a method of triggering that minimises the error between the different modules. The total time it takes for a triggering instruction to travel between server and client can be written as

$$T = T_s + T_m + T_r + T_a$$

.

where T_s is the send time, i.e. the time it takes from the moment when the user has given the trigger command until the message has left the network device. T_m is the time the message travels in the network medium between two network adapters. T_r is the receive-time: the time it takes from the moment when a message has arrived at the network interface until it gets delivered to the application by the operating system. Finally, T_a is the time it takes for the application to react to the message, the application time.

Elson et. al. observe that a broadcasted message in a wireless network arrives at virtually the same time at all receiver interfaces [4]. This means that errors introduced in the system mostly stem from differences in receive and application times between modules. Both application time and to some extent the receive time are dependent on operating system processes such as interrupt handling and scheduling, and are thus hard to predict.

The straightforward synchronisation approach is to assume both of these times to be the same on all modules and start the recording as soon as a triggering message arrives at the application.

Another, more intricate approach allows us to get rid of the receive time altogether: If we make sure that the clocks of the laptops are synchronised by using a time synchronisation daemon, a future start-time can be sent instead of a direct triggering message. All servers wait until the specified time and then start the recording. In this manner we avoid errors introduced by the network altogether. Only the error of the time daemon must be taken into account instead.

We have implemented both methods and found them to work equally well, as both the time daemon error and T_r seem to be small compared to T_a.

The current version of the software captures video to local memory during recording because the bandwidth of the laptop hard drives is not reliable enough for direct streaming. After a capture, the raw data is losslessly compressed to disk. On our laptops with 1 GB of RAM, we can capture approximately 80 seconds of video with a resolution of 640x480 pixels at a rate of 15 frames per second. Given the scheduling dependent constraints of a non real-time operating system such as Windows 2000, we have targeted the triggering of the system at this capture rate. This proves to be a good tradeoff between memory space and recording speed, allowing us to record a fair amount of video while still maintaining a frame rate comparable to studio systems such as [10].

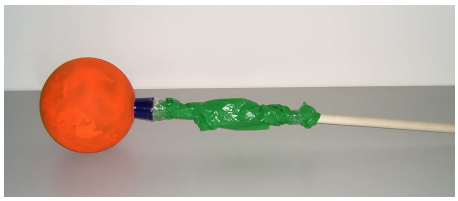

Figure 4: *The marker object that is tracked through the scene to establish 2D correspondences between camera frames.*

Calibration. To accommodate general camera setups, we use an approach that does not require a common view of a calibration object for all cameras. Instead, a *virtual* calibration object that covers the working volume is constructed over time by tracking an easily identifiable object [1, 2]. We mainly follow the approach of [2]. However, we lift the constraints that the cameras must be registrable by triangulation from the base camera pair and that the working volume must be dark during calibration.

Intrinsic calibration is done independently for each camera using Tsai's method [12]. This is reasonable since our cameras do not allow for varying internal parameters. This requires us to record an image of a checkerboard for each camera, from which the internal parameters can be computed. However, the checkerboard does not have to be visible from all cameras simultaneously.

At the place of recording, we first obtain a number of 2D correspondences via tracking the marker object depicted in Fig. 4. The midpoints of the marker object are determined and the correspondences are used to robustly compute the relative position and orientation of each camera pair [15, 7].

Since the pairwise relative position of the cameras can be found only up to a scale factor, a global registration has to be performed to achieve global calibration. This is done by constructing a graph $G = (V, E)$ that represents the relationship between cameras, Fig. 5. The cameras $C_i \in V$ are the vertices, and known relative position and orientation between any pair of cameras $(C_i, C_j) \in E$ are represented by the edges of the graph. The graph is undirected, since known relative position and orientation $(\mathbf{R}, \mathbf{t})$ for camera pair (C_i, C_j) also determines the relative position and orientation of (C_j, C_i) as $(\mathbf{R^T}, -\mathbf{R^T t})$.

Once the graph is set up, it is searched for cyclically connected subsets $S_k \subseteq S$ containing all cameras. The set $\bar{S} := S \setminus (\cup_k S_k)$ consists of all cameras not belonging to any cycle. If $S = \bar{S}$, no cycles exist and a special treatment is necessary. Figure 5 illustrates a situation where all sets are nonempty.

The unknown scale factors for the pairwise translations are determined for each S_k independently. This is done by solving an over-determined linear system of equations. The equations correspond to cycles in the graph and require that the scaled translations along the traversed path have a total sum of zero. That means the cycle represents a closed curve in three-dimensional space. Using these partial registrations, parts of the virtual calibration object (i.e. the path of the tracked marker object) are reconstructed. The remaining step is to register all subsets S_k and all $C_i \in \bar{S}$ with each other to form a globally consistent calibration.

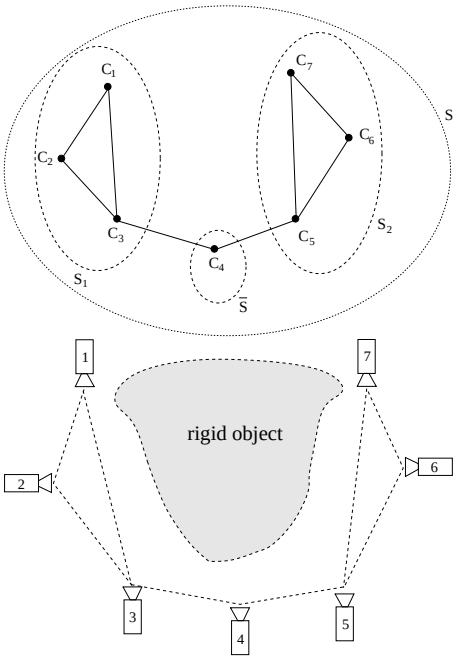

Figure 5: *An example graph with non-empty sets $S, S_k, \bar{S}$ (upper image) and a corresponding scene where this situation could occur (lower image).*

In this framework, the improvement of our method can be stated like this: The restriction of the vertices V being interconnected by three-cycles in the graph G as required by the algorithm of Chen et. al. [2] is lifted and arbitrary connectivity of the graph G is allowed for as long as the graph is not disconnected.

APPLICATION EXAMPLE

To demonstrate the usability of our system, we recorded and reconstructed a craftsman at his working place. For this purpose, we connected two cameras to each laptop, using a total of eight cameras. All equipment fitted easily into a normal car. We set up the system in about two hours, including recording of calibration and calor correction sequences. One of the camera setups used for recording is shown in Fig. 6. During the period of two hours, we acquired about 30 short sequences with three different camera configurations. Preparation of a different camera configuration takes about 20 minutes and requires the recording of calibration sequences. Packing together the camera modules after finishing the recordings took another hour.

Figure 6: *A camera setup for the recording of a copper engraving craftsman.*

Figure 7: *Two of the eight input views for one of the sequences (upper row) and some reconstruction results with virtual table and background (lower row).*

That leaves us with five hours of work for recording 30 sequences with three different setups.

For 3D - reconstruction of the sequences we used the algorithm presented in [5]. We give a short outline here. At first we generate silhouettes of foreground objects using background subtraction based on image statistics [3]. Additionally, shadow pixels are classified and removed based on large difference in intensity, but only a low difference in hue. To reconstruct the volume, we intersect the back-projections of the foreground silhouettes to approximate the visual hull [9] of the objects in the form of a cubic, binary voxel volume. The single voxels of the volume are projected onto the camera planes and checked whether they are inside the silhouette. If this is the case for all cameras observing the voxel, then it is marked as belonging to the reconstructed volume.

For rendering from novel perspectives, we attach a billboard (a small rectangle with the same orientation as the image plane of the novel view) to each voxel center and texture it with the original images. Texture coordinates are obtained by projecting the corners of the billboard rectangles into the images. The two cameras that are nearest to the novel view are used for texturing, and the resulting textures are blended according to the angle between their cameras viewing directions and the novel viewing direction.

Results of this algorithm applied to the craftsman sequences are shown in Fig. 7.

SUMMARY

We have presented a portable system for multi-video recording, intended for capturing scenes not suitable for indoor-studio recording. By eliminating all cable connections between the cameras, we have constructed a modular system which allows for higher portability and view configuration.

The setup consists of a number of physically unconnected modules that form an ad-hoc network to function as one system operated from a client application. A module is lightweight (less than 10 kg), has an online time of about three hours and can be transported in a backpack. The current performance is 80 seconds of video in 640x480 pixels resolution captured at 15 fps.

The biggest contributors to a synchronisation error are the application reaction time and, in some cases, the receive time. Both of these error sources could be greatly reduced, allowing for higher capture rates, by porting the software to a real-time platform. We plan to do this as a future step of our project.

The modular design makes scaling of the system easy: Just add a new module to the setup by placing it at the desired recording position. Once started and running, it becomes a part of the multi-video capturing environment.

We extended existing methods for external camera calibration using virtual calibration objects in order to make them applicable to a broader range of camera setups.

With todays available methods and active research in the fields of image-based rendering and computer vision, we believe that there are many areas of application for recording systems as the one described here. In the near future we hope to test our system by doing on-site recordings in theatres, at sports events and in the workshops of craftsmen. After all, it might be time to come out of the studio and capture the real world.

REFERENCES

[1] A. Azarbayejani and A. Pentland. Camera self-calibration from one point correspondence. Technical Report 341, MIT Media Lab, 1995.

[2] X. Chen, J. Davis, and P. Slusallek. Wide Area Camera Calibration Using Virtual Calibra-

tion Objects. In *Proceedings of CVPR*, volume 2, pages 520–527, 2000.

[3] K. Cheung, T. Kanade, J.-Y. Bouget, and M. Holler. A real time system for robust 3D voxel reconstruction of human motions. In *Proc. of CVPR*, volume 2, pages 714–720, June 2000.

[4] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*, 2002.

[5] B. Goldluecke and M. Magnor. Real-time, free-viewpoint video rendering from volumetric geometry. In T. Ebrahimi and T. Sikora, editors, *Visual Communications and Image Processing 2003*, Proceedings of SPIE, pages 1152–1158, June 2003.

[6] R. Hartley and A. Zisserman. *Multiple View Geometry*. Cambridge University Press, 2000.

[7] B. K. Horn. Recovering baseline and orientation from essential matrix. http://www.ai.mit.edu/people/bkhp/papers/essential.pdf, January 1990.

[8] T. Kanade, H. Saito, and S. Vedula. The 3D room: Digitizing time-varying 3D events by synchronized multiple video streams. Technical Report CMU-RI-TR-98-34, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, December 1998.

[9] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan. Image-based visual hulls. In *Proceedings of ACM SIGGRAPH*, pages 369–374, 2000.

[10] C. Theobalt, M. Li, M. Magnor, and H. Seidel. A flexible and versatile studio for synchronized multi-view video recording. In *Proc. of Vision, Video and Graphics 2003*, pages 9–16, 2003.

[11] C. Theobalt, M. Magnor, P. Schueler, and H.-P. Seidel. Combining 2D feature tracking and volume reconstruction for online video-based human motion capture. In *Proceedings of Pacific Graphics 2002*, pages 96–103, 2002.

[12] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–334, August 1987.

[13] I. Ulrich, C. Baker, B. Nabbe, and I. Nourbakhsh. IEEE-1394 digital camera windows driver. http://www-2.cs.cmu.edu/~iwan/1394/, August 2002.

[14] B. Wilburn, M. Smulski, K. Lee, and M. Horowitz. The light field video camera. *SPIE Proc. Media Processors 2002*, 4674, Jan. 2002.

[15] Z. Zhang. A new multistage approach to motion and structure estimation: From essential parameters to euclidean motion via fundamental matrix. Technical Report RR-2910, INRIA, June 1996.