Silhouette-Extraction for interactive VR-Systems involving dynamically changing rear-projection screens

Jörn Herwig, Sven Thüring, Alfred Schmitt

University of Karlsruhe
Institut für Betriebs- und Dialogsysteme
Am Fasanengarten 5
76128 Karlsruhe
Germany
Phone +49 (0)721 608-3965
Fax +49 (0)721 608-8330
E-mail : {herwig, thuering, aschmitt}@ira.uka.de

Abstract: Markerless optical motion tracking is the method of choice if you don't want to use expensive tracking hardware or invasive equipment in a VR-System. Combined with silhouette-extraction based on background subtraction real-time interaction is reachable. But most methods using this scheme are most likely unable to adapt their background model to the fast changing unpredictable images shown on a projection screen. This restriction makes them unusable for applications involving spatially immersive displays (SID) where users are interacting in front of projection screens.

This paper describes an approach to solve this challenging problem. Based on a client-server architecture it makes use of frame buffer and screen timing information delivered by the presentation computer and standard digital cameras attached to the tracking computer. The system requires no special hardware; therefore it is comparatively cheap and furthermore fast and reliable. Sufficiently accurate algorithms are used, allowing usage in a real-time system.

Key words: Virtual Reality, background subtraction, dynamic background, silhouette extraction, markerless optical tracking

1 – Introduction

The presented system is developed out of the need for a fast and reliable yet cheap user tracking system. It is used to provide input for the interactive VR system involving dynamics simulation of mechatronic systems described in [1].
No special tracking-hardware or invasive equipment obstructing the mobility of the user should be required. Because of these requirements we decided to build up a camera-based tracking system. The system will base on silhouettes captured from multiple viewpoints which will be used in a later step to perform a 3D reconstruction [2]. This will allow the recognition of the user pose and gestures.

Currently the most promising method for silhouette determination is background segmentation mainly because of execution speed issues. Additionally this scheme offers a wide range of different approaches fitting various needs.

Unfortunately a last requirement prescribed by our system unveils a severe limitation. This requirement looks simple at first sight: the user is acting in front of a screen lit by a rear projection. First we give a short overview of available segmentation methods. This will help to explain the consequences resulting from this requirement.

There are numerous approaches of figure–ground segmentation. A brief introduction of various methods is given in [3]. Especially the use of the background subtraction methods, based on a static background model, is very popular. This approach especially fits indoor scenes without fast changing light conditions and a nearly static background. Another way of segmentation uses a statistical method to rate the input images. Those methods give better results in scenes involving small background changes e.g. moving tree leafs.

But none of these various methods is able to handle scenes showing a projection screen. No background model adaptation is fast enough to cope with the rapidly changing images shown on the screen. This often leads to full classification of the projection screen area as foreground (see Figure 1). This makes the results completely unusable for further processing. Conclusion of this restriction is the limited coverage of the working space in front of the projection screen, as seen in Figure 2.
So instead of adapting the model to the changing conditions it has to imply all possible states. This is only possible with the knowledge of the currently shown screen contents.

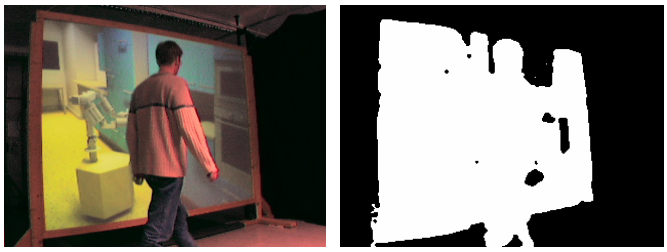

Figure 1 : Projection screen detected as foreground.

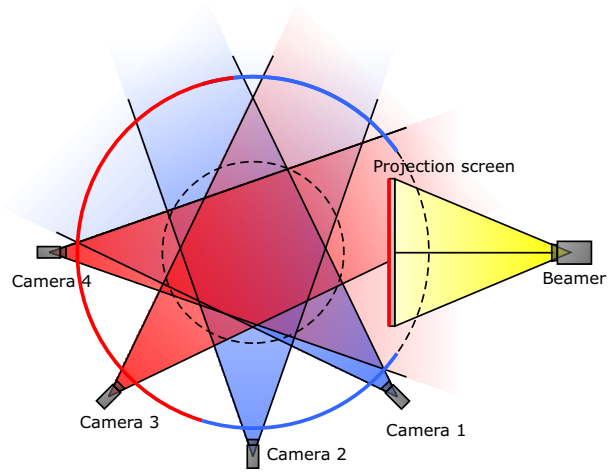

Figure 2 : Limited coverage of working space.

The presented method therefore uses a digital screenshot of the screen content to build an intermediate background model fitting a conventional background subtraction algorithm. The intermediate model construction is based on a global background model covering all possible conditions.

The reference implementation is integrated into an existing program called Visor. Visor already handles the segmentation of a static environmental background [1].

2 – Related Work

In the field of 3D reconstruction from multi-camera video there are already a lot of papers. A review of methods for volumetric scene reconstruction is given in [4]. But the combination of 3d video acquisition methods and projection display methods are still quite seldom. The blue-c system [5], developed at the ETH Zürich, uses active visible light in combination with synchronized shutter glasses and an adaptive silhouette method to subtract the user from the dark background of the projection screen. Another approach implemented in the SIDEshow system [6] uses infrared light illuminating a screen behind the user, in combination with a black and white camera in front of the user, which shows only the black silhouette of the human. Moeslund et al. [7] are using s-video cameras and a priori knowledge about the scenario in the camera's field of view. A magnetic tracker calculates the 3D position and orientation of the user's head. They suppose that the background is brighter than the user, and the skin has a good reflectance for long wavelengths. Since they are just interested in the pointing direction, they threshold the red channel to find the position of the largest objects, the head and the hand, to calculate the 3D direction of the hand. Hirose et al. [8] proposed a system to segment the user in a VR-CUBE from the background to generate a video avatar. They used infrared cameras to deal with the poor light conditions and simulate a static reference background image which is then subtracted from the infrared image containing the user. The simulation of the background also gives information about the illumination the user is exposed to. This could be used, e.g. to estimate an intensity threshold for segmenting the user.
All approaches use a special hardware configuration to overcome the problem of finding a silhouette. This leads to restrictions like losing the texture information or the huge technical effort in the case of the blue-c system.

3 – Overview

3.1 – System

The current paper only addresses the silhouette detection inside the area of the screen. The minimal system (Figure 3) for this task consists of the following components

- a tracking computer with a digital camera
- a beamer attached to a presentation computer
- the rear projection screen
- and a network connecting both computers.

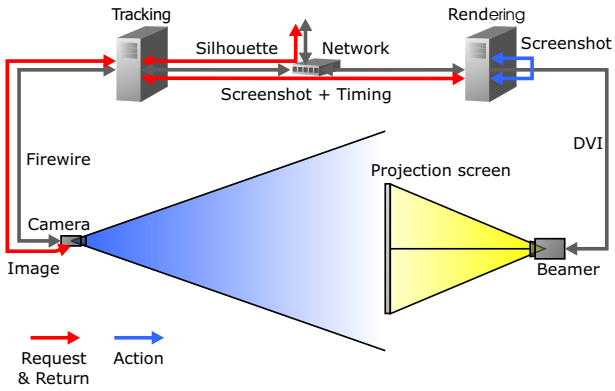

Figure 3 : Configuration.

3.2 – Work flow

The work flow follows a general scheme:

- Beginning with a request from outside of the core system or alternatively a periodic timer the tracking computer starts the segmentation process.
- The tracking computer demands the current screenshot and timing information from the presentation computer (see chapter 7 – Image synchronisation).
- Using the timing information the tracking computer triggers the camera to capture an image.
- The screenshot delivered by the presentation computer is first transformed geometrically to match the projection screen (see chapter 4 – Geometrical transformation).
- The result is processed by a colour transformation to adopt to the colour appearance of screen (see chapter 5 – Colour transformation).

- Finally the actual segmentation takes place (see chapter 6 – Segmentation).
- The resulting silhouette is returned.

4 – Geometrical transformation

After the screenshot from the presentation computer was received it has to be brought into correspondence with the image of the projection screen delivered by the camera. To reach this the whole mapping of the screen contents onto the camera image has to be replicated. This incorporates every transformation beginning with the projection by the beamer up to the projective mapping by the camera.

4.1 – Simplifications

Based on the given properties of the configuration and the necessary data it's possible to make some simplifications. With the help of these it is possible to reduce the complexity of the necessary calculations.

The following conditions are assumed:

- The projection of the beamer is nearly free from distortion.
- The image captured by the camera is nearly equivalent to a perspective projection. This can be assured by a preceding camera calibration which is indispensable for the later on volume cut anyway.

For the succeeding steps only the pure mapping is of interest, because the complete processing works on the two-dimensional image data. Thus an exact reconstruction of the position in space isn't necessary.
Taking this into account the following additional generalizations result:

- Small deviations from the above ideal properties are acceptable.
- The focal distance of the camera is unimportant.
- The distance of the projection screen from the image plane is irrelevant.

Continuing with these limitations and generalizations the mapping can be determined with the following sub steps.

4.2 – Detect screen

The first step is to detect the area which is occupied by the projection screen within the camera image. This area is used as a mask later on to eliminate unwanted side effects in the environment.

To detect the projection screen some full screen black and white images are displayed by the presentation computer. Both image sequences get averaged to minimize noise.
The absolute difference between both images is calculated. After an adjacent raise in contrast a border detection takes place using a Canny filter. The biggest contour according to the surrounded area is chosen from the recognized outlines. This contour is further on used as the boundary of the projection screen.

4.3 – Grid division

To allow a sufficiently exact reconstruction even if distortion appears the projection screen is split into single grid cells. These cells are afterwards calculated independently. If the cell decomposition is sufficiently fine global distortion can be ignored and the mapping inside a single cell can be assumed to be a perfect perspective projection.
The necessary granularity depends on the degree of distortion and the image resolution. Usually 6-12 rows and 8-16 columns are needed.

The splitting is achieved by displaying a checkerboard pattern containing the desired number of cells, as shown in Figure 6. Three circles are used to detect the orientation of the projection screen. Most suitable for the image data analysis is the brightness information, e.g. the Y channel of the YCrCb colour space..

4.3.1 – Corner detection

First the corner points of the grid cells have to be detected. Any sufficiently accurate algorithm allowing sub-pixel accuracy can be used here. Our implementation is based on a combination of minimal eigenvalues used for course detection and saddle points for refinement.

The unchanged input data contains interfering characteristics in the environment. Additionally an insufficient differentiation between the brightness values across the projection screen area exists. Because of this results aren't satisfactoring at first go.
To increase the quality of the source input the previous captured black and white images are used. An almost completely fade out of the environment and a spread to full contrast range can be achieved. Processing is done according to formula 0 where w_0 is the original white image, b_0 the original black image, i the input image and o the output image.

$$\begin{aligned} d_0 &= w_0 - b_0 \\ w &= w_0 \cdot d_0 \\ b &= b_0 \cdot d_0 \\ d &= w - b \\ o &= \frac{\min(i \cdot d_0, w) - s}{d} \end{aligned} \tag{1}$$

Result of this operation is a nearly perfect checkerboard pattern with high contrast and very few disturbing pixels in the environment (Figure 4). Now the corner point detection achieves much better results.

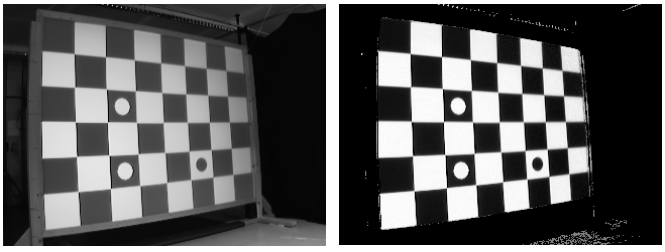

Figure 4 : Optimized checkerboard.

To make the following steps easier the border and the inner points are detected independently. To do this for the border points the search is limited to a stripe around the projection screen contour. An inverse display of the pattern is used so that the two corner points of the black corner cells can be found too. The already found points are left out in the mask when doing this.
For the inner points the stripe is removed from the complete mask of the area.

4.3.2 – Surrounding polygon

After the inner and outer points are found separately the surrounding polygon including all border points can be calculated. It is used as source of the following assignment of all points to their respective grid cells.

The initial step is to detect one of the upper corner points. The uppermost of the two is preferred. Starting from this point the order of border points is determined.

Following criteria are used

- distance between the points
- the scalar product

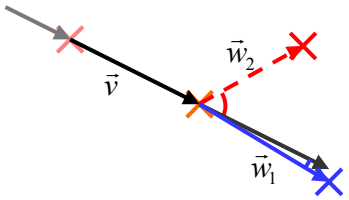

Figure 5 : Order of polygon points.

$$\min(|\vec{w}| \cdot (2 - \vec{v} \cdot \vec{w})) \quad (2)$$

At the same time the angle can be used to define the four corners of the projection screen.

$$|\vec{v} \cdot \vec{w}| < \varepsilon \quad (3)$$

This information also allows refining the choice of the starting point.

4.3.3 – Point-grid assignment

The separate detection of inner and outer points simplifies the assignment of four points each to a cell considerable.
Originating from the known upper corner point the grid is processed row by row. To complete the first cell inner point with minimum distance to the corner point has to be detected. This point lies diagonal across the cell. Together with the next and previous point in the surrounding polygon these points form the first cell. The determined point is removed from the list of inner points. This way the point is ignored during the following distance comparisons.
The second cell uses the next point of the polygon as reference point. In this case the next polygon point and the previous diagonal point complete the grid cell. See Figure 6 for an example.

This process can be continued till the first row is completed. After this it is applied to all remaining rows, too.

It is possible to construct cases where this method fails. This can happen because a wrong inner point is lying closer to the reference point. Under real circumstances including nearly quadric grid cells and avoiding low viewing angles this will hardly happen. But other methods providing point-grid assignment e.g. [9] have to deal with these limitations, too.

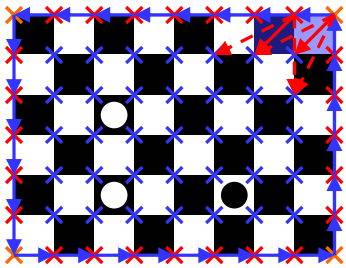

Figure 6 : Calibration pattern and point-grid assignment.

4.3.4 – Orientation

In a final step the orientation of the screen has to be detected. This is necessary to handle rotated or even mirrored projections.

This is done using the circle marks. The three inverse circles construct a orthogonal triangle. The horizontal distance is three cells, the vertical distance only two cells. This information can be used to set the orientation.
To simplify the search the brightness values in the centre and around ¾ between centre and one corner are compared. If both have a minimum absolute difference a circle mark is found.

Circles are used because they don't interfere with the previous corner detection steps.

4.4 – Perspective projection

At this time all four points for each cell are known. If the grid division is sufficiently fine the mapping can be treated as a perfect perspective projection as already mentioned above. Furthermore it can be assumed that one of the points lies on the image plane. So it is equal with the mapped point. And at last the focal length of the camera is set to 1.
All this simplifies the following equations.

4.4.1 – Transformation

Based on the circumstances the following order of transformations can be set up. They allow the calculation of a closed mapping starting from a single cell ($P_1 - P_4$) inside the screenshot. Result of this operation is the image of the cell ($B_1 - B_4$) as seen by the camera. The transformations are illustrated in Figure 7.

① 2-dimensional transformation so that a point p is equal to the origin
② general rotation and scale to form the desired orientation
③ 2-dimensional transformation so that p is equal to its mapped point
④ perspective projection

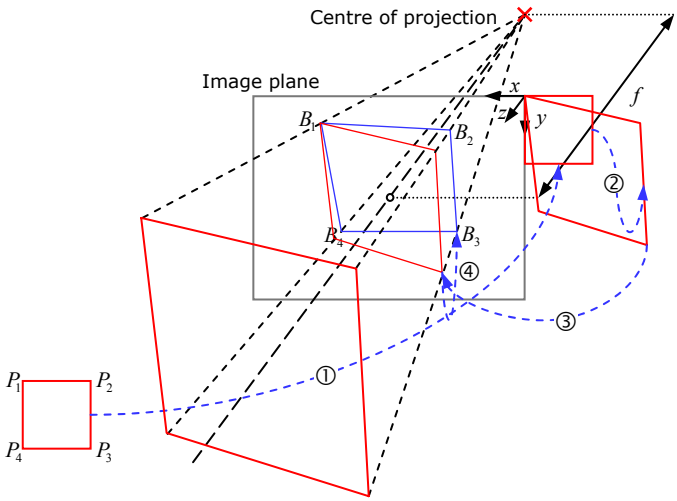

Figure 7 : Mapping transformation.

4.4.2 – Calculation

Written into formulas this results in (4).

$$P_i = \begin{bmatrix} x_i \\ y_i \\ 0 \end{bmatrix}, \quad B_i = \begin{bmatrix} u_i \\ v_i \\ 0 \end{bmatrix}$$

$$T = \underbrace{\begin{bmatrix} 1 & 0 & 0 & u_1 \\ 0 & 1 & 0 & v_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{translation (mapping)}} \underbrace{\begin{bmatrix} t_{11} & t_{12} & t_{13} & 0 \\ t_{21} & t_{22} & t_{23} & 0 \\ t_{31} & t_{32} & t_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{rotation and scale}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{translation (origin)}}$$

$$= \begin{bmatrix} t_{11} & t_{12} & t_{13} & -t_{11}x_1 - t_{12}y_1 + u_1 \\ t_{21} & t_{22} & t_{23} & -t_{21}x_1 - t_{22}y_1 + v_1 \\ t_{31} & t_{32} & t_{33} & -t_{31}x_1 - t_{32}y_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

$$T_p = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 1 \end{bmatrix}}_{\text{projection}} \cdot T$$

$$= \begin{bmatrix} t_{11} & t_{12} & t_{13} & -t_{11}x_1 - t_{12}y_1 + u_1 \\ t_{21} & t_{22} & t_{23} & -t_{21}x_1 - t_{22}y_1 + v_1 \\ 0 & 0 & 0 & 0 \\ \frac{t_{31}}{f} & \frac{t_{32}}{f} & \frac{t_{33}}{f} & 1 + \frac{-t_{31}x_1 - t_{32}y_1}{f} \end{bmatrix}$$

This can be solved towards the t_{ij} resulting in a mapping of the screenshot source pixels onto their equivalents in the camera image.

4.4.3 – Gauss filtering

Applying a Gauss filter to both the transformed screenshot and the camera image improves segmentation results. On the one hand still existing minor distortions can be eliminated almost completely. And on the other hand the camera noise is reduced.
It showed up that filtering based on the screenshot coordinate system provides the best results.

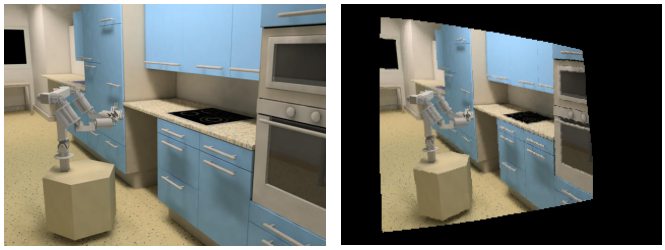

Figure 8 : Geometrical transformation.

5 – Colour transformation

Now that screenshot and camera image are coincident the second obvious problem shows up. A clear difference in colour appearance can be spotted making direct use in the actual segmentation impossible.

5.1 – Colour distortion

On the way from the graphics card memory of the presentation computer to the usable camera image a large amount of colour distortion occurs, e.g. when the CCD-chip of the camera captures the scene. To make it even harder the difference depends on the position on the screen and therefore in the camera image, too. All these things considered lead to the knowledge that an arithmetic replication can't be covered with a single closed formula.

The following processing is the core of the complete procedure. Only this allows a usable segmentation under the given circumstances.

5.2 – Requirements

Before our approach is presented which solves this problem the taken choice is explained with the following list of requirements and the resulting consequences.

- Suitable for real-time usage
  The method can't be too complex.
- Position dependant
  A calibration run to collect measurement data has to be done.
- Compromise between calibration afford and accuracy
  Different quality settings should be possible.

5.3 – Approach

A method matching the required characteristics is the trilinear interpolation of samples spread across the entire colour space.
Each colour channel is divided into equidistant segments. Thus the complete colour space is divided into a grid. According to the desired accuracy and the required speed this can be improved. It turned out that a subdivision into five segments or in other words six samples is a good compromise.

5.3.1 – Calibration

During the calibration phase a loop runs through all grid points showing a full screen display of the corresponding colour combination on the projection screen. Therefore a sample for each combination of the subdivided colour channels is captured.
In this case not only the averaged image is stored but also the minimum and maximum for every pixel is detected. Beyond that the maximum change of two successive images is stored. This additional data is necessary for the later segmentation.

5.3.2 – Interpolation

During the interpolation the previous collected calibration data is used to calculate the distorted colour values.
The input image, which in most cases is the already geometrically transformed screenshot, is processed per pixel. To minimize the runtime it is recommended to pass through those pixels only that display the projection screen.
According to the input colour value the corresponding grid cell can be determined in which the colour is located. The eight surrounding knots contain the samples that act as source data of the following interpolation. To be more precise not the complete images are used but rather the values at the exact location of the input pixel.

Trilinear interpolation: The easiest approach to calculate the result from the eight present source values is a simple trilinear interpolation.

At first the relative position of the input colour inside the grid cell is calculated. Normalized onto the range [0, 1] it can be used as weight of the individual source samples.

The linear interpolation has depended on its principle drawbacks in case of strong nonlinearities. An example for this are strongly saturated colours which result in almost maximum values delivered by the camera although the saturation hasn't reached its maximum.

Spline interpolation: The shortcomings of the linear interpolation can be faced by taking the saturation of the resulting colour values above the input values into consideration. The weight used to interpolate isn't the linear factor any more which is given by the relative position. Instead the precalculated value dependent on the input value is used.

During the first stage an iteration runs across the complete scale of each channel. The values of the two other channels are set to a fixed value. These depend on the requirements of the used colour model. When using the HSV model S and V are fully saturated while the H curve is detected to work on the pure colour hues. To determine the values of the brightness channel V, H can be any value and S is set to zero. This way only unsaturated grey shades passed through.
The average value of the projection screen area is calculated for every sample image. Together with the source value it forms the coordinates of one point. Then all points at once are interpolated with a cubic B-spline curve.
Using the de Boor scheme allows the calculation of the interpolating spline. This spline can now be used to set the normalized weights of every input value in relation to both surrounding samples. Figure 9 shows an example.

The use of spline interpolation leads to a noticeable better approximation of the colour transformation compared to the captured images. Thus the segmentation results are improved.

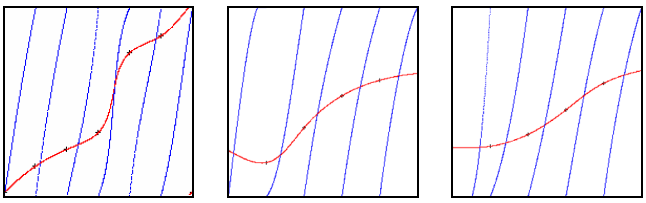

Figure 9 : Spline interpolation using HSV model.

To get even better results an additional calibration run can be performed. This one includes more shades of the colour values relevant for the spline interpolation. This way the approximation gets closer to real values once more.

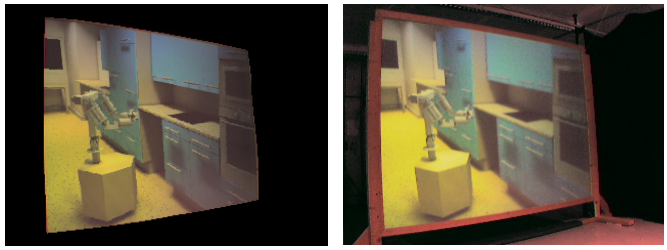

Figure 10 : Colour transformation compared with real image.

6 – Segmentation

After the screenshot is fitted to the image captured by the camera concerning shape and colour appearance the actual segmentation can be performed. Every single pixel runs through a thresholding process assigning it to the foreground or background.

First only the area of the screen is segmented. The result of this segmentation is passed over to the segmentation of the environment. This one ignores the screen area resulting in a segmentation of the complete image.
After that some filter stages follow which improve the quality of the silhouette. These remove disruptive pixels or take the last detected silhouette into consideration during the rating. The resulting silhouette can be used as input data for skeleton fitting or calculating the visual hull [2].

6.1 – W4 segmentation

The so called W4 segmentation [10] was chosen for the reference implementation. Reasons for this are on the one

hand the low calculation costs allowing usage in a real-time system and on the other hand the good results in the already existing environment segmentation.

The meaning of the term W4 (Who?, When?, Where? and What?) already shows the initial goal of this method. The recognition of persons and their actions.
W4 is based on a static background model. It is gained through the analysis of a pure static background image sequence. The minimum and maximum during this sequence is detected for every pixel and channel. Additionally the maximum change between two successive images is saved.
Minimum and maximum form an interval which is first enlarged by the maximum change (Figure 11). This interval is used to classify the request pixels. If the value of the request lies inside the interval it is rated as background otherwise as foreground.

- i – colour channel
- c_i – colour values of calibration
- d_i – colour value request
- ε – maximum change

$$\forall i : c_{i,\min} - \varepsilon \le d_i \le c_{i,\max} + \varepsilon \qquad (5)$$

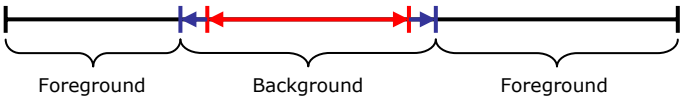

Figure 11 : W4 segmentation.

6.1.1 – Cyclic channels

Special attention has to be paid to cyclic channels e.g. the hue of the HSV colour space (Figure 12). These channels do not have a minimum and maximum. Instead an overflow occurs when reaching the maximum value.
This characteristic has to be taken into account during determination of extreme values as well as tests for interval inclusion. Otherwise we receive bad or even completely unusable silhouettes.

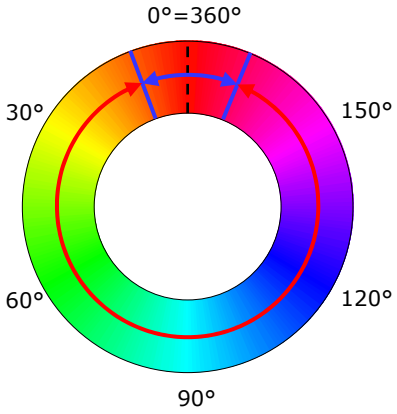

Figure 12 : Cyclic channels.

6.2 – Extension

We extended our algorithm to solve the given task. Whereas the original variant is based on a single minimum-maximum pair the extension incorporates up to multiple hundred pairs. There is exists one pair for each captured colour sample. As already mentioned coverage takes place during the calibration phase of the colour transformation. The maximum change now is of global validity.

Before an input value is classified the involved minima and maxima of the current pixel are interpolated as described for the colour transformation above. The resulting interval can be used with the W4 method as usual.
All extremes together form an intermediate background model of the W4 algorithm.

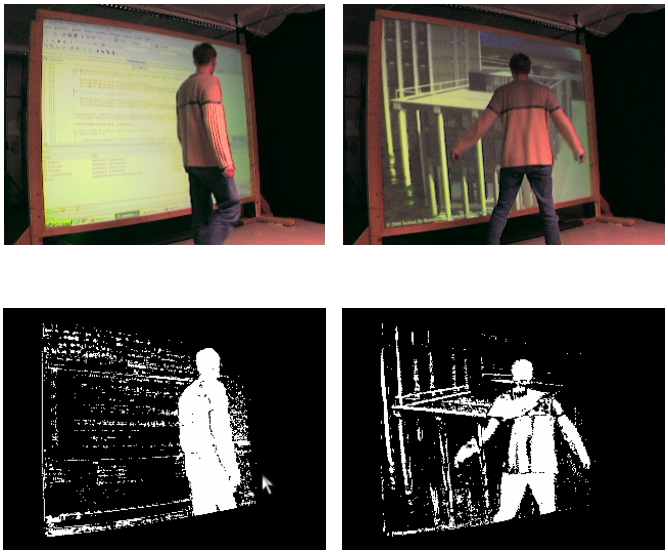

Figure 13 : Unfiltered segmentation results.

7 – Image synchronisation

Before the segmentation can start input images are required. The screenshot is requested from the presentation computer and is delivered in perfect quality. It might seem that the creation of the camera image looks unproblematic. The camera is ordered to capture the image at a given moment and a little bit later the image is available.

7.1 – Mixed images

But if the camera image is taken at specific time the content on the screen can be a mixture of two images. Sometimes a visible dividing line exists. Above the line the new screen content appears, below the old one is still visible (see Figure 14). This behaviour called tearing is especially frequent when fast scene changes occur. Strong contrasts intense the appearance of this effect. Screenshots containing such artefacts are completely unusable for the presented segmentation method.

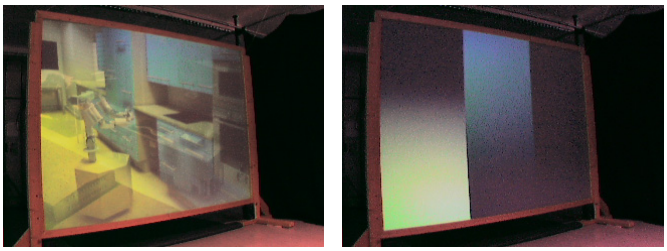

Figure 14 : Mixed images.

7.2 – Scanline refresh

The defects are caused by the screen refresh type of the beamer. Even if driven digitally, e.g. using the DVI interface, LCDs and beamers refresh the screen one row at a time. This type of refresh is also referred as scanline refresh. Usually the refresh starts at the top of the screen and continues downwards. Initially scanline refresh was invented to be used with cathode-ray tubes (CRTs). It is even essential for this type of display to allow presentation of raster images. Because of compatibility reasons it is still in use today.
If the camera capture doesn't take at the exact moment of a finished screen refresh mixed images occur.

7.3 – Approach

An additional characteristic of the scanline refresh offers an approach to solve this problem.
After the image is completely displayed no new row will be refreshed during a short period of time. This short break called vertical sync is required when working with CRTs. It allows to reset the deflection of the cathode-ray to the upper left corner. After that the next refresh cycle is able to start. See Figure 15.

If the camera captures an image during this period of time the image should be almost defect free. Some minor artefacts remain because of the inertia of the liquid crystals used in the beamer and the shutter of the camera.

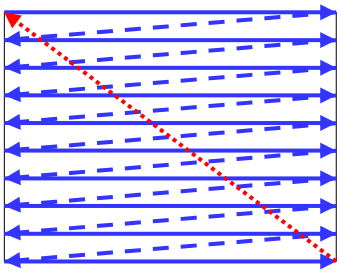

Figure 15 : Vertical sync.

Unfortunately the available period of time is very short. Using a resolution of 1024 columns and 768 rows displayed at a rate of 60 Hz results in a vertical sync time of only about 0,77 ms.
To complicate things even more direct access to timing information isn't available to the tracking computer without the use of special hardware. Even if the vertical sync pulse could be detected things couldn't be improved. Triggering the camera is always tainted with latency. And this latency usually lasts very long compared to the vertical sync.

But the presentation computer has the ability to request the currently displayed row. Knowing the row number it is possibly to calculate the time remaining till next sync. If the tracking computer is capable of receiving this information across the network the maximum latency can be determined. The total latency consists of the network runtime, the trigger latency of the camera and the code runtimes. Synchronized image capture becomes possible if the total latency is known. Varying network latencies can be compensated by detecting the maximum latency during a test run.

7.3.1 – Determination of the latency

Without consideration of the latency synchronization works like shown in Figure 16

- Tracking computer sends request
- Presentation computer detects current display row
- Remaining time till sync is calculated

This remaining time allows the triggering of the camera simultaneous with the vertical sync.
In case the latency is known it's possibly to check whether the remaining time is sufficient. If this isn't true the time of one or multiple complete refresh cycles can be added to it. The adjusted remaining time is sent back to the tracking computer.

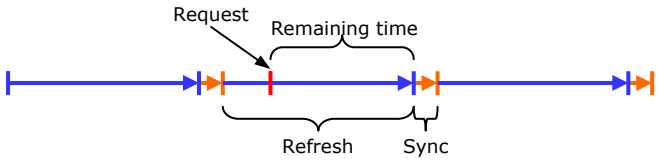

Figure 16 : Remaining time.

The screenshot can be taken at any time during screen refresh. While the vertical sync is active this isn't possible because the frame buffer is replaced during this period. If a postponement is necessary because of the latency the screenshot capturing has to be delayed until the beginning of the appropriate cycle. Otherwise it can be captured directly after returning the remaining time.

Bar pattern: The synchronization makes use of a special bar pattern. The screen is divided into three white bars each of equal width. They are shown on a black background. The high contrasts allow a reliable detection of the refresh position. Again the contrast range can be improved with the method presented during the geometrical transformation.

To begin with the bar on the left side is shown during several cycles. This is done to make sure a stable image is visible. Now the screen is switched to the middle bar. This one is shown for exactly one refresh cycle. The directly succeeding image shows the right bar.
Thus the middle bar is completely visible only during the vertical sync. Before the left bar is visible below the refresh position and afterward the right bar appears above it. The current refresh position can be detected according to this transition.

Determination of refresh position: To detect the transition position a sweepline method is used. The sweepline runs across the screen area twice beginning at the top. This search

is based on the screen coordinate system thus the geometrical transformation has to be detected before. During the first pass the left and middle bar are considered. During the second pass these are the middle and right one. This way the sweepline follows the chronological activity.
As criteria indicating the refresh position the switch-over in gradient direction is used. The gradient is calculated from the difference between the intensities averaged across the complete bar width. To reduce errors caused by position imprecision the border areas are omitted.
Because of noise there can exist multiple switch-overs in gradient direction inside actually homogenous areas. To handle this the switch with the highest change rate is picked (Figure 17).
In the beginning the gradient direction is set towards the middle bar. This allows detection of the immediate change caused by the visibility of the middle bar.

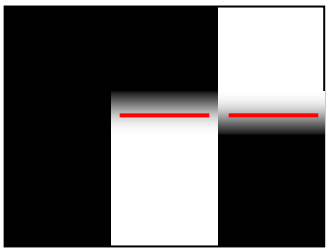

Figure 17 : Refresh position.

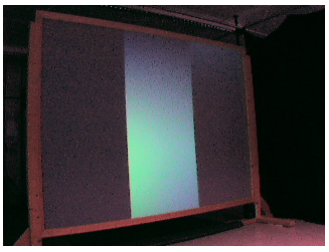

Figure 18 : Synchronized image.

Knowing the refresh rate and the sync duration it is possible to translate the refresh position into the necessary latency change. Originally the position exists as row number. It has to be normalized to fit into the range [0, 1]. In case the transition occurs on the left image half this value is decreased by 1. Finally the distance to the desired state is obtained.
This value simply has to be multiplied with the refresh time and subtracted from the current latency.
Afterwards a newly pass is performed to verify the current changes or adopt them again. The process finishes successfully in case multiple subsequent passes show stable results around the desired ideal situation like in Figure 18. As mentioned before a perfect image is very unlikely. To cope with this only a fall short a certain limit is required.

8 – Cost and performance

Because solely standard components are used the hardware cost are only marginal compared with other tracking devices. This offers the chance to use such technics where a tight budget won't allow the application of expensive equipment. In this manner new input possibilities become available, offering extended ways of user interaction. Thus an intensified immersion experience even for small systems is at hand.

The minimal system with only one camera and tracking computer already allows usage of the silhouette extraction for two-dimensional user tracking and gesture recognition. Multiple cameras make 3D reconstruction through volume cutting possible.

The current unoptimized reference implementation calculates 3-6 frames per second running on a generic 3.0 GHz computer. Optimization hasn't took place mainly because the main goal at the time is the proof of concept. Because of this variability took priority before speed, allowing comparison of various colour spaces, interpolation types and other parameters.
Nevertheless during development a less feature rich version already ran at around 15-20 frames per second. This throughput should allow usage in a real-time system.

9 – Conclusion and future work

We presented a set of methods allowing background segmentation of scenes showing a rear-projection screen. Based on this the usage of markerless optical user tracking in spatially immersive display environments becomes available. This offers an additional way of user interaction with the system resulting in an intensified experience of immersion.

The core elements can be easily adopted for use with a wide range of background segmentation algorithms. This way the algorithm of choice can be extended to support visible projection screens.

9.1 – Future work

The current base still offers many capabilities of improvement.

9.1.1 – Partially visible screen

At the moment only fully visible projection screens are supported. Because of this the possible camera positions are still restricted.
Detecting partial visibility is simple because the detected screen contour touches the image border. Handling it is harder. The cell decomposition has to be adaptive, refining subdivision in partially visible grid cells. Because the available resolution is limited the decomposition has to stop if it reaches a given limit. In this case it stops because of the missing points.
To achieve this only the geometrical transformation needs to be changed, all other parts remain untouched.

9.1.2 – Update background model

Currently the global background model is static during complete runtime. Because of this the method is unable to adopt itself to changing light conditions. If the illumination differs from the calibration state the segmentation results become worse.
Goal of this extension should be the ability to adapt to slowly changing lighting conditions. No specific solution for updating the background model is available at this time.

10 – References

[1] Finkenzeller D., Baas M., Thüring S., Yigit S. and A. Schmitt. VISUM: A VR System for the Interactive and Dynamics Simulation of Mechatronic Systems. Virtual Concept 2003, Biarritz, 5.-7. November 2003.

[2] M. Fautz. Objekt und Texturrekonstruktion mit einer robotergeführten Kamera. Ph.D. Thesis., Universität Karlsruhe (TH), Fakultät für Informatik, October 2002.

[3] Moeslund T. B. and Granum E. A survey of computer vision-based human motion capture, Computer Vision and Image Understanding, v.81 n.3, p. 231-268, March 2001.

[4] Dyer Charles R. Volumetric scene reconstruction from multiple views. Foundations of Image Understanding, Editor L. S. Davis, Kluwer, p. 469-489. Boston, 2001.

[5] Gross M., Würmlin S., Naef M., Lamboray E., Spagno C., Kunz A. and Koller-Meier E. Blue-C: a Spatially Immersive Display and 3C Video Portal for Telepresence, ACM Transactions on Graphics (TOG), Volume 22 issue 3, Association for Computing Machinery, New York, NY, July 2003.

[6] Davis J. W. and Bobick A. F. Sideshow: A silhouette-based interactive dualscreen environment. Technical Report 457, MIT Media Lab, August 1998.

[7] Moeslund T. B., Störring M. and Granum E.. Vision-Based User Interface for Interacting with a Virtual Environment. 9th Danish conference on pattern recognition and image analysis. Copenhagen, August 2000.

[8] Hirose M., Ogi T. and Yamada T. Integrating live video for immersive environments. IEEE MultiMedia, Volume 6, Issue 3. p. 14-22, July 1999.

[9] Shu C., Brunton A. and Fiala M. Automatic Grid Finding in Calibration Patterns Using Delaunay Triangulation. Technical Report NRC-46497/ERB-1104, August 2003.

[10] Haritaoglu I., Harwood D. and Davis L. W4: Who, when, where, what: A real time system for detecting and tracking people. In Proceedings of the Third Face and Gesture Recognition Conference, p. 222--227, April 1998.