Bachelorthesis
Detecting people with a simple webcam

Vincent van Megen (0513482)

November 12, 2008

Contents

Chapter 1

Introduction

Imagine that you could not see the difference between cars and the streets they move on. Crossing the street would be quite problematic. Luckily, visually identifying objects is an easy task for humans; we can almost instantly tell the difference between two objects. For a computer, however, detecting or identifying objects has proven to be a difficult task [PP00]. A special form of visual object detection is people detection, which can be seen as object detection with only two classes of objects: human and non-human.

1.1 Why visual people detection?

There are many possible applications for a system that can automatically detect humans using vision. All over the world, camera's are used to monitor or detect people. Although in most of these cases, people detection is done by a human staring at a video-screen. A computer program that can automatically identify humans on camera images would be very useful in these situations [FBFC+97]. A computer never looks away from the screen, and it never gets bored or tired.

A system for detecting people becomes even more useful when it is coupled to a robot. For example, after the attack on the twin towers on 9/11, a robot was used to search the rubble for survivors [Bur04]. Robots are useful here because they can reach places humans can not. Also, sending a robot instead of a person into such a dangerous situation, could potentially save lives of rescue workers. This robot, however, was still operated and monitored by a human. If the robot would search for survivors on it's own, and only bother the rescue workers until it had found a human, more rescue workers would be available for other tasks. Another example would be a robotic car [HlZkQh06]. Such a car would need to know where the pedestrians are located, or else it might run them over.

There are other methods of people detection available to these systems (sonar for example). So why use vision? First of all, because it is cheap. Camera's are very cheap these days, especially low-end camera's such as webcams. If you could use these camera's to detect people, you would have a very cheap human-detection sensor.

Another reason would be the wealth of information in the visual spectrum. With sonar for example, you can detect where a person is in space, but not where this person is looking. When you use visual people detection, it might be possible to see where a person is looking.

1.2 The difficulties of people detection

Unfortunately, visually detecting humans is a complex problem. First of all there are the problems associated with normal object detection. Changes in lighting condition can lead to significant changes in the picture that needs to be processed by the system, and shadows can be a problem too (after all, they often have exactly the same shape as the object that needs to be identified). Occlusion is also a big problem, it is hard to identify an object if you cannot see it completely. In order to identify an object in such a situation it would need to infer what is behind the occlusion, or recognize the visible parts of the object [MSZ04].

People detection can be even harder, since people are all different. We have different shapes, (skin) colors, and can wear clothing that can distort our shapes even more. On top of that, we can take on different poses, and move in almost completely unpredictable ways. With all these problems combined, people detection is an extremely difficult task.

1.3 People detection with a webcam

Despite these difficulties, vision remains an excellent way to detect humans or objects. However, are these cheap (web)camera's good enough to accurately detect objects? Or even a human? In this thesis we will try to answer the following questions:

1. Is it possible to detect motion using a simple webcam?

2. Is people detection possible, using such a webcam?

These questions are answered by implementing a computer program that tries to detect people, using a cheap webcam. First, we explain other requirements on the method proposed in this thesis in Chapter 2. Then several ap-

proaches to detecting people from the literature are reviewed in Chapter 3. From these methods, we chose the following method: shape based people detection. We try to implement this method by using a neural network, explained in Chapter 5. But in order to use this neural network, motion detection needs to be done, which is explained in Chapter 4. Finally, we discuss our results and suggest further reseach in Chapter 6.

Chapter 2

System requirements

The shape based people-detection method proposed by this thesis is developed at the request of a company called R2R. This company has created a robotic mannequin. Their plan is to sell the mannequin to shops, who can place her behind their shop window. The mannequin can display certain kinds of behaviour, such as taking on different poses. This would attract more attention to the shop, and create a more dynamic showcase. The mannequin has several layers of behaviour, with higher layers being able to interupt the lower layers. The detection of people will be quite high in the hierachy, so the standard behaviour of the robot can be interupted if the system detects a person standing in front of the shop window. So, for example, if the system detects someone standing in front of the shop window, it would stop doing it's standard poses and start to wave to that person.

Since the company is looking for a system that can actually be used, computational complexity is very important. If the system cannot work in real-time, the mannequin would start waving long after the customer in front of the shop window had walked away. Not only would this look silly, it would defeat the whole purpose of the system. Computational complexity is not only a factor because of the speed, but also because of the costs. With a highly complex system, a powerful computer is needed. The more powerful the computer, the more expensive it will be. And since R2R plans to sell the system, keeping the production costs as low as possible is very important.

The system also needs to be able to deal with multiple people at the same time. Since the system will be located in a shopping street, the chance of several persons walking into view at the same time is quite high. Luckily, the system does not need to identify people. It does not need to know which specific customer is standing in front of the shop window, it only needs to detect that the person is standing there. It would, however, be a plus if the system would be able to track people in it's field of view. In this case it would be able to wave to a person walking by.

A disadvantage of the system being located in the shopping street is that a shopping street is a highly dynamic sitation. This means that the lighting conditions will change often, along with the background and possibly even the position of the camera. The reflection of the glass of the shop window would also be a problem, since it would be hard to discern between the reflections and the real-world events outside. Distance is also a factor. Most systems discussed in the literature focus on surveillance camera's, which are located above the target, at a reasonable distance. Since R2R plans to incorporate the camera in the mannequin itself, the distance to the target would be quite small.

Also, it would be preferable if the system was easy to upgrade. At first R2R is looking for human detection system, but eventually tracking might be built into the system. So to summarize, the system must be:

- Cheap
- Reliable
- Easily upgradable
- Able to operate in real time
- Able to deal with outdoor situations (changing lighting conditions, etc)
- Efficient at short distances
- Able to handle multiple people at the same time

In the next chapter several methods of people detection from the literature are discussed, and a choice for a certain method will be made based on these criteria.

Chapter 3

Methods of people detection

The image captured from the camera first needs to be processed in order to do people detection: the background needs to be separated from the foreground. This is crucial because this enables the system to detect objects that are of interest (especially in people detection). If, for example, there is a painting hanging on the wall opposite to the camera, we do not want the system keep detecting it as an object. Once this distinction is made, the real people detection can begin. In the literature there are three basic forms of people detection. These will be discussed in the next subsections.

3.1 Detection by shape

The first method of people detection is based on the shapes of the foreground objects. First the foreground picture needs to be divided in different objects. This can be done by edge detection. Once the different objects have been established, their precise shape can be processed in order to determine if it's a human shape or not. The simplest example of this is template matching [CC05]. Another example is the dispersibility, which is calculated out of the height, width, perimiter and the area of objects [HlZkQh06]. To increase performance, the system can also look at the shape of different parts of the body. So instead of looking at the shape of the entire body, one could look at the shape of the lower part (the legs), the centre (torso), etc. [MSZ04, NTG+06]. Another way to look at the shape of the body is by using wavelets [PP00].

3.2 Detection by motion

Besides having very distinctive shapes, humans also move in a specific way. Their speed is quite distinctive [VMBP97]. The way they move can also dis-

tinguish humans from other objects. For example, if an object is detected, you can compare the amount of motion in it's lower half to the amount of motion in the upper half. If the object moves more in the lower half, it probably is a human, since humans move their lower half (their legs) more than their torso when they are walking. The disadvantage of this method is that it is computationally quite complex, which leads to lower processing speeds. Also, it can be seen as an extension of shape-based detection, since in order for the method to track the speed of an object, it needs to identify an object.

3.3 Face detection

Detecting humans by detecting their faces is based on a very simple truth: Every human has a face. So it is fairly safe to assume that as soon as you detect a face, you have also detected a human. Detecting faces can be seen as a special form of shape detection, since separate objects need to be identified first. These objects can then be identified as a face [Sal04].

The advantage of detecting humans in such a way, is that it provides a very useful framework for extra functionality, such as gaze tracking. The problem with face detection is that the system does not only need to discern between the foreground, background and different objects on the foreground, but also on different parts of the objects. Luckily, our heads are almost always the highest point of our body, so this provides a certain clue as to where the face would be. But even then, face detection is a complex problem.

3.4 Choosing the method

At the end of Chapter 2, several requirements were mentioned. A very important one of these was speed. The system would be completely useless if it could not operate in real time. Shape-based people detection fulfills this requirement better than the other two, especially since shape-based detection methods with speeds up to 11 frames per second have already been implemented [HlZkQh06, LSAD03].

As mentioned above, detection based on faces or motion also rely on shapes to get their first information. So implementing a motion-based detection system would still require implementing a shape-based detection system. And a shape-based detection system could ofcourse be upgraded to include motion or face recogntion, if it is required.

So the method proposed in this thesis will use shape-based people detection. However, before people detection can begin, the foreground needs to be discriminated from the background. This will be discussed in the next chapter.

Chapter 4

Foreground detection

The camera that is used by the system is a Trust WB 3400T, with a hardware resolution of 640 x 480 and a framerate of 30 frames per second. The software of the system is implemented on a PC, using windows XP. The actuall program is written in Java. The choice to use Java was made because of the Java Media Framework (JMF). This Java API is used to avoid having to program image-processing features, which are readily available in the JMF. By using this API, more time was available to work on the actual task of people detection, instead of losing time on image-processing.

However, some image-processing was still needed. Although the JMF handled the first stages of the image processing (capturing the feed from the camera, computing the color values, etc.), the end result of this processing was a long array of color values. These color values represented the different pixels in the frame.

In order to increase the speed of the system, the resolution was lowered to 320 by 240. This will drastically increase the processing speed. A large amount of information, however, might be lost due to this choice.

4.1 Differential motion analysis

Differential motion analysis means that a reference frame is captured, and incoming frames are compared to this reference frame. If there are any changes in the values of the pixels, it probably means that there was motion at the location of that pixel. The reference frame is renewed for every 5 frames so the system can deal with shifts in camera position.

Once a difference value is calculated for a pixel, it is compared to a threshold. This treshold is used to prevent small changes in color being detected as

motion. If the difference between the value of the current pixel and the value of the pixel in the reference frame is larger than the treshold, the pixel is marked as being in motion.This differential motion analysis was done in greyscale, since comparing pixel values is a lot easier if there is only 1 value for each pixel. The RGB model was transformed into greyscale using formula 4.1

$$\text{Greyscale} = \frac{\text{red} * 0.6 + \text{green} * 0.4 + \text{blue} * 0.1}{3} \tag{4.1}$$

Implementing differential motion analysis showed that such a simple form of motion detection was not sufficient. The first problem was the automatic contrast of the camera. This meant that if the overall lighting conditions changed (for example, if someone were to stand in front of the camera) the camera adapted the contrast to adapt to this new situation. However, since the camera detects motion by detecting changes in color, the difference in contrast also led to the detection of motion where there was none. The changes caused by automatic adaptation were so severe, that the adaptation would lead to higher difference values than actual motion.

Luckily, this problem was easily solved. The automatic adaptation of the camera could be turned off by the software that came with the camera. Unfortunately there is no way to turn it off using the JMF, so it will have to be turned off manually on each computer that the system will be used on. Or a camera without automatic adaptation will have to be used.

Further testing then revealed that this was not enough to eliminate noise. Due to slight vibrations (and just noise from the camera), motion would still be detected where there was none. Although the amount of noise was significantly lower than before automatic adaptation was turned off, it was still too much to create an accurate threshold. So if a reference frame was taken, and the camera vibrated a little, it would see motion along all edges, for the next 4 frames.

4.2 Statistical background estimation

In order to deal with these problems, a new method for separating the foreground from the background was used, called statistical background estimation [MG03]. The statistical background estimation used in this thesis is a relatively simple one, yet many modifications exists which may or may not increase performance [SBR00].

In statistical background estimation, instead of taking 1 reference value for each pixel, every 5 frames, it computes the mean for that pixel over an extended

period of time. This makes calculating the difference value a lot more reliable, since a pixel on an edge (for example on an edge between greyscale values 6 and 8) would take the mean value (in this case 7).

Also, instead of greyscale, the color values are now used in calculating the difference value. Although this leads to a drawback in the speed of the system, the results with differential motion analysis led us to believe that too much information was lost in the greyscale transformation.

Equation 4.2 expresses the mean color value. X_k is the value of a pixel, with X_{k-4} being the value of this pixel 4 frames ago. This calculation was done for each of the components (red,green, and blue). In our implementation, k was set to 5.

$$\text{ColorMean} = \frac{X_k + X_{k-1} + ... + X_1}{k} \tag{4.2}$$

Comparing these means to the pixel value of the current frame was done using the euclidian distance between the mean RGB values of a pixel and the current RGB values of that pixel. This can be seen in equation 4.3.

$$\text{Distance}(new, colormean) = \sqrt{(R - R_{mean})^2 + (G - G_{mean})^2 + (B - B_{mean})^2} \tag{4.3}$$

But even with this new adaptation, it would be hard to chose an accurate threshold. If the threshold would be set too high the system would become insensitive to motion. If the threshold would be set too low, vibration would be a problem. In order to deal with this, statistical background estimation uses a new way of calculating the treshold.

Instead of taking a fixed value (such as 6 or 8), the system keeps track of the last few values that a certain pixel had. The standard deviation of these values is then calculated, and the distance between the current pixel values and the mean pixel values is compared to the standard deviation of that pixel. The formula for the standard deviation can be seen in equation 4.4.

$$\sigma = \sqrt{\frac{\sum_{x=1}^{k} \text{Distance}(x, colormean)^2}{k - 1}} \tag{4.4}$$

If the difference between the current pixel value and the mean pixel value was more than 2 times this standard deviation, it would mark that pixel as being in motion. The result can be seen in Figure 4.1. The pixels that are seen

Figure 4.1: Empty frame without noise reduction

Figure 4.2: Motion detected when walking past the camera

as the foreground are marked red. The resulting motion values with someone walking past the camera can be seen in Figure 4.2.

Since this was still quite a large amount of noise, region-based noise reduction was implemented. This basically meant that the resolution was lowered, with each block of 4 by 4 pixels forming a single pixel in the new frame. The new pixel had the mean value of the 16 old pixels as it's value. So effectively this lead to a resolution of 80 by 60 pixels.

This new method of noise reduction proved to be quite effective. The results can be seen in Figures 4.3, 4.4, and 4.5 where the pixels marked as foreground are now colored blue.

Figure 4.3: An empty frame with noise reduction

Figure 4.4: A walking human with noise reduction

Figure 4.5: Motion detected from a desk-chair

Chapter 5

The neural network

As can be seen in Figures 4.3 to 4.5, the statistical background estimation and the noise reduction results in a clear distinction between the foreground and the background. This distinction is good enough to start detecting humans.

Because the distinction between different patterns of motion has to be made by the system, the choice was made to use a neural network. Presenting the new 80 by 60 frame to the network directly was not a reasonable option, since this would lead to 4.800 input nodes. Training a network with so many nodes would take ages, so the motionmap was rescaled again, to 16 by 12, this time taking the sum of the processed pixels as their new value. This meant that a new frame was saved of a 16 x 12 resolution, with each pixel having a value between 0 and 400. And these 16 x 12 frames were used to train and test the neural network. For a summary of all preprocessing steps, see Figure 5.1.

There were three stages in testing the network. The first stage is to see if it could accurately detect motion. So it would only need to discern between an empty frame and a frame with a moving object in it. The second stage is to see if it could discern between an empty frame, a frame with a human, and a frame with another kind of moving object in it. The final stage is to test the network in an outdoor situation.

5.1 Motion detection

In this test, the results of the foreground detection and abstraction (Arrays of 16 x 12 binary values) are saved. These patterns were then manually reviewed, and 125 empty patterns were selected, along with 125 patterns with a moving human in the frame. Another 25 empty and 25 human patterns were selected to be used as the test set. The network was trained for 500 epochs, with 5 hidden units. This lead to a performance of 100% on the train- and test set. Since this

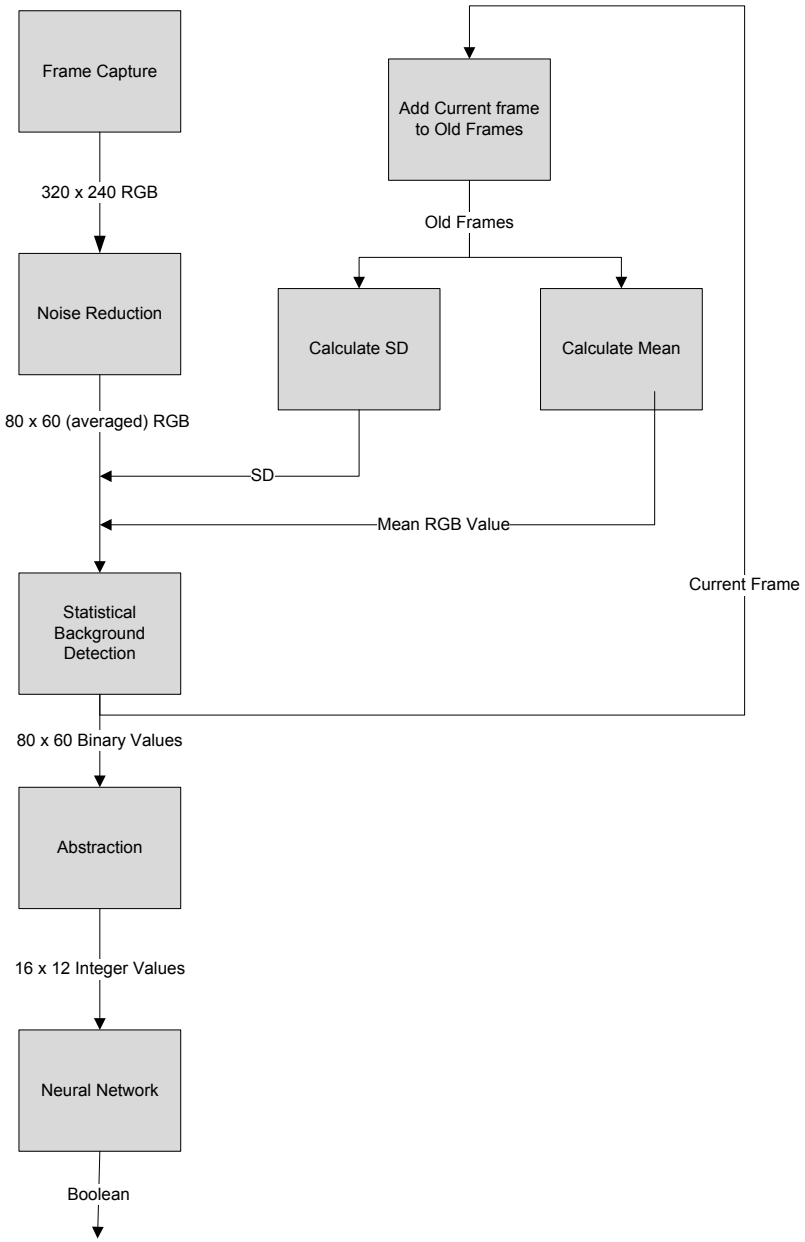

Figure 5.1: The inner workings of the system

performance was already perfect, no further testing was done.

5.2 Human detection

In this test, 50 new patterns were added to the 250 patterns in the trainset. These patterns contained a moving desk-chair. Another 25 of these chair-patterns were added to the test-set. The network was then tested in different configurations. The results of these tests can be seen in Table 5.1.

Table 5.1: Human detection results

Number of hidden	Epochs	Train	Test
5	500	94,33%	97,36%
	1000	99,66%	98.68%
	2000	100%	98,68%
10	500	93%	97,36%
	1000	99,66%	98.68%
	2000	100%	98,68%
20	500	92.66%	97,36%
	1000	99,66%	98.68%
	2000	100%	98,68%

As can be seen in Table 5.1, the performance of the network on the test set does not increase when training more than 1000 epochs. There is also no difference between networks with 10 and 20 hidden units, so increasing the size of the hidden layer even more will most likely have no effect.

5.3 Outdoor performance

Since the previous tests were done on manually selected samples, in an indoor situation, they are not nearly as hard as the task the network would have to perform if it were incorporated in the mannequin of R2R. In order to test the network performance on a more realistic situation, the system was taken outside to capture patterns on a busy shopping street. Motion analysis results of this shopping street can be seen in Figures 5.2 and 5.3.

These patterns were then manually classified into human and non-human, but no frames were removed in either the train- or the test-set. The train set consisted of 750 patterns, and the test set of 250 patterns. The result of this third stage is presented in Table 5.2.

Figure 5.2: Person detected on a shopping street

Figure 5.3: Motion caused by a car

Table 5.2: Results of outdoor testing

Number of Hidden	Epochs	Train	Test
5	500	86.95%	94.09%
	1000	88.4%	92.12%
	2000	89.88%	92.51%
	4000	92.0%	91.73%
10	500	87.46%	93.70%
	1000	88.0%	93.70%
	2000	89.06%	92.91%
	4000	90.26%	91.73%
20	500	87.2%	93.70%
	1000	87.86%	92.91%
	2000	89.06%	92.51%
	4000	90.26%	90.55%

While the results from Table 5.2 seem very good, there is a problem. Of the 1000 frames taken in an outdoor situation, most of them contained cars or cyclists. In only a few of them actual pedestrians could be seen. So the network could theoretically reach a very high performance if it constantly claims that there is no human in the picture. In order to check if this was happening or not, a confusion matrix was computed of the best-scoring network. Since multiple networks have the same, highest score of 93.70%, one had to be chosen. The choice was made to compute the confusion matrix of the network with 10 hidden units, trained for 1000 epochs. The resulting confusion matrices can be seen in Tables 5.3.

Table 5.3: The confusion matrices

Set		Empty(target)	Human(target)
Train	Empty(output)	638	82
	Human(output)	7	23
Test	Empty(output)	233	15
	Human(output)	2	4

This reveals that the network only manages to detect humans in about 20% of the frames with a moving human in them, which is a significantly lower number than the overall network performance would indicate.

Chapter 6

Conclusion

6.1 Discussion

In this thesis an attempt was made to create a people-detection system using a simple webcam and a neural network. The first challenge was to detect objects (or motion) using the webcam. Our results show that this can be done perfectly with a cheap webcam, with a performance of 100%. The next task was to discern between an empty frame, a frame with a human in it, and a frame with a deskchair in it. This can be done using a simple webcam, with a performance of 98.86%.

The real challenge, however, was detecting humans in an outdoor situation, in a stream of frames. While the network seems to have a fairly high performance of 93.70 %, the confusion matrix revealed otherwise. The system only detected about 20 - 25% of the humans present in the frames.

So in our system, the webcam and the motion detection performed quite well. However, interpreting the motion data coming from the camera turns out to be a bit harder, in an uncontrolled outdoor situation. Most likely, the problem is the features presented to the neural-network. In the implementation described in this thesis, a simple low-resolution version of the motion-data was presented to the network. This is a very crude and simple feature. If more processing was done before presenting the data to the network, the performance might increase dramatically.

The drawback of this pre-processing would be the speed of the system. The more processing needs to be done by the system before presenting the data to the network, the slower the overall process becomes. Since real-time performance is of such importance, a balance would need to be found between the number and complexity of the features (and thus the performance of the network), and the speed of the overall system.

Another problem that was revealed in the tests, was the distance requirement. The webcam needs about 4 meters to get a human completely (from head to toe) in the frame. However, in the case of the mannequin used by R2R, the distance would most likely be 1 meter or so.

So, to come back to our introduction and system requirements: is it possible to detect people with a simple webcam? It would seem that this is most certainly the case. The system proposed in this thesis, however, does not fullfill all requirements of R2R. While the system is cheap, reliable (no stability issues were found during testing, although no expressive reliability tests were done), easily upgradable (adding new features is especially easy), able to operate in real time, and able to deal with outdoor situations, it is not able to detect people at short distances, and it could do a lot better at detecting multiple people at the same time. This last issue, however, could be resolved with extra features.

So, all in all, it is possible to detect people using a simple webcam, however, it is not yet possible to do so at the short distance that R2R requires. If R2R wants to use the system, a different lens would need to be installed on the camera, or another camera with a wider angle of view would be needed.

6.2 Further research

6.2.1 Color model

In Chapter 4, the choice was made to use the Euclidian distance to compare RGB values. Humans, however, judge differences in color in a different way [Lam94]. There are color models that model human perception of color differences, such as cieLAB or CIE XYZ. We have tried implementing a version of cieLAB to see wether it would have any effect on the performance of the system. Unfortunately, the model turned out to be too complex to implement in the scope of this bachelor-thesis. It might be interesting to see if different color models have an effect on the performance of a system such as the one proposed in this thesis.

6.2.2 Object segmentation

It might be interesting to separate the different objects in the frame, and present these to the network separately. This could lead to a whole array of new features, such as the height, centroid and overall area of an object. But more importantly, object segmentation might make it possible to track objects over several frames, so an object's speed could be computed. The speed of an object is a very useful feature, since anything going faster than 8 km/h can be discared as a walking human, which would make the distinction between humans and all kinds of motorized vehicles a lot easier.

6.2.3 Network memory

If the network detects a human in a certain frame, it is most likely that it will do so again in the next frame. With neural networks, it is possible to use the output of a previous frame as input for a new frame, giving the network some form of memory [Elm90]. By doing this, no information is "lost" between two frames, which might increase performance of the network.

6.2.4 Detecting motionless humans

In this thesis, the whole method of detecting people relies on motion to be detected first. However, if a person were to stop and look at the mannequin of R2R, within a few seconds the system would no longer "see" this person. Although this can partially be solved by looking at a larger selection of frames with the statistical background estimation, this will always fail if the person remains still long enough. There are methods that do not use motion information to search for humans or faces [HAMJ02]. Although these systems have drawbacks of their own, it would be very interesting to couple such a system to the one proposed in this thesis.

6.3 Acknowledgements

I would like to thank Richard, Peter and Axel of R2R for their guidance during my internship and Franc Grootjen for his guidance in writing this thesis.

Bibliography

[Bur04] S. Burion. Human Detection for Robotic Urban Search and Rescue. *master's thesis, Robotics Inst., Carnegie Mellon Univ*, 2004.

[CC05] Carlos Castillo and Carolina Chang. An approach to vision-based person detection in robotic applications, 2005.

[Elm90] J.L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[FBFC+97] J.A. Freer, B.J. Beggs, H.L. Fernandez-Canque, F. Chevrier, and A. Goryashko. Automatic intruder detection incorporating intelligent scenemonitoring with video surveillance. *European Conference on Security and Detection*, pages 109–113, 1997.

[HAMJ02] R.L. Hsu, M. Abdel-Mottaleb, and A.K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):696–706, 2002.

[HlZkQh06] C. Hao-li, S. Zhong-ke, and F. Qing-hua. The Study of the Detection and Tracking of Moving Pedestrian Using Monocular-Vision. *Lecture Notes in Computer Science*, 3994:878 – 885, 2006.

[Lam94] J.M.G. Lammens. *A Computational Model of Color Perception and Color Naming*. PhD thesis, State University of New York, 1994.

[LSAD03] M. Leo, P. Spagnolo, G. Attolico, and A. Distante. Shape based people detection for visual surveillance systems. *Lecture Notes in Computer Science*, 2688:285–93, 2003.

[MG03] Ekinci Murat and Eyup Gedikli. *Background estimation based people detection and tracking for video surveillance*, volume 2896. Springer Berlin / Heidelberg, 2003.

[MSZ04] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human Detection Based on a Probabilistic Assembly of Robust Part Detectors. *Lecture notes in computer science*, 3021:69–82, 2004.

[NTG+06] A. Negre, H. Tran, N. Gourier, D. Hall, A. Lux, and J.L. Crowley. Comparative Study of People Detection in Surveillance Scenes. *Lecture notes in computer science*, 4109:100 –108, 2006.

[PP00] C. Papageorgiou and T. Poggio. A Trainable System for Object Detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[Sal04] Al-Shehri Saleh. *A simple and Novel Method for Skin Detection and Face Locating and Tracking*, volume 3101. Springer Berlin / Heidelberg, 2004.

[SBR00] J. Sullivan, A. Blake, and J. Rittscher. Statistical Foreground Modelling for Object Localisation. *Lecture notes in computer science*, 1843:307–323, 2000.

[VMBP97] Patrick Vannoorenberghe, Cina Motamed, Jean-Marc Blosseville, and Jack-Gerard Postaire. *Automatic pedestrian recognition using real-time motion analysis*, volume 1311. Springer Berlin / Heidelberg, 1997.