Computers and Electronics in Agriculture 76 (2011) 169–174

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

Development of a real-time computer vision system for tracking loose-housed pigs

Peter Ahrendt*, Torben Gregersen, Henrik Karstoft

Aarhus School of Engineering, Dalgas Avenue 2, 8000 Aarhus, Denmark

ARTICLE INFO

Article history:
Received 8 July 2010
Received in revised form
10 December 2010
Accepted 21 January 2011

Keywords:
Loose housed pigs
Real-time computer vision
Tracking
Location

ABSTRACT

Changes in regulations for livestock animals will in the near future call for loose-house pig breeding systems. These new systems will increase the workload for the farmers, as location and identification of animals will require more time than before. This paper presents a real-time computer vision system for tracking of pigs in loose-housed stables. The system will ease the workload for farmers in identification and locating individual animals. The system consists of a camera and a PC. The PC runs a tracking-algorithm, estimating the positions and identities of the pigs. The tracking algorithm operates in 2 steps. The first step builds up support maps, pointing to preliminary pig segments in each video frame. In the second step the support map segments are used to build up a 5D-Gaussian model of the individual pigs (i.e. position and shape). The system has software correction for fisheye distortion coming from the camera lens. The fisheye lens allows the camera to monitor a much larger area in the stable. The algorithms are developed in MatLab, implemented in C and runs in real-time. Experiments in the lab and in the stable demonstrate the robustness of the system. The system can track at least 3 pigs over a longer time span (more than 8 min) without loosing track and identity of the individual pigs in a realistic experiment.

1. Introduction

In the future, laws and regulations for animal keeping will force the farmers to use loose housing systems. Pigs are gregarious animals, and by using loose housing systems the welfare of the pigs is increased. It is assumed that the stress of each pig will be decreased and the quality of the meat will be increased (Damm, 2008). The farmers workload will increase too as it is predicted that the farmer will spend more time to find pigs for checks or treatment. In loose housed systems the number of pigs is typically larger than 50. Therefore reliable equipment that can help the farmer to identify and observe the animals will be necessary in the future. It is preferable to implement the equipment with an easy-to-use, graphical interface on a mobile unit, and ensure that employees from different countries can easily use the systems. By entering the ID of the pig on a mobile unit the position of the pig could be displayed on the graphical display on a mobile unit and the pig could be illuminated by a light source, so it is easy to find.

Several different approaches have been investigated in the past to solve such tracking problems. Examples are GPS positioning, Bluetooth- or WiFi-networks and RFID-based approaches for cows (Huhtala, 2007). However, practice shows that it is more problematic to attach electronic devices to pigs than to cows. For instance, cows can carry a larger weight and they have less physical contact among each other. Among these techniques, RFID is one of the more promising for pigs. Passive RFID equipment could be mounted in the ear of each pig and a number of RFID-readers could be placed in a lot of different places in the stable (Tan et al., 2008). However, this leads to a large number of RFID readers/antennas which is an expensive solution. Instead of passive RFIDs, active RFIDs could be mounted in the ear of each pig and RFID-readers should only be placed at a few positions in the stable (Ni et al., 2004). This can increase the position accuracy, but unfortunately the active RFIDs consume a lot of power, so it will be necessary to change or charge batteries quite often. The active RFIDs can also easily be too heavy for the pig. Furthermore the pigs will typically spoil the equipment or remove the equipment from each other. Therefore this solution will not reduce the farmers workload.

Our approach has instead been to develop a camera-based computer vision system for tracking the loose housed pigs. The advantage of a computer vision solution is that it is not necessary to mount expensive, battery consuming RFIDs on the pigs as described above and it is a non-contact solution.

Problems of course also occur with the camera solution in such a harsh environment as a pig barn. Disturbances include dust and dirt which to some extent can be alleviated by automatic calibration of the camera software, but also ammonia vapors which necessitates an air tight camera enclosing to protect the electronics.

* Corresponding author. Tel.: +45 60 63 66; fax: +45 87 30 27 31.
E-mail address: pah@iha.dk (P. Ahrendt).

0168-1699/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.compag.2011.01.011

The computer vision approach has been used for many years in a multitude of areas such as poultry tracking (Sergeant et al., 1998; Leroy et al., 2006) and tracking of rodents and insects to monitor behavior and activity (Noldus et al., 2002; Noldus et al., 2001). In Cangar et al. (2008), real-time image analysis is used to monitor and track calving cows. Detailed tracking of positions, area and body orientation was achieved at a framerate of 1 Hz. Pigs have also been tracked in many previous attempts. In Tillett et al. (1997) the focus is the tracking of individual pigs and finding their specific movements such as bending of the back and nodding of the head. Shao and Xin (2008) use pattern recognition methods on real-time videos to assess the thermal condition of group-housed pigs. Perner (2001) analyzes grayscale videos to determine pig position and movement in the stable by subtraction of a reference background image, thresholding and segmentation with the line coincidence method. Image-subtraction algorithms are also the basis for the tracking of pig behavior in Lind et al. (2005). Here, the system is thoroughly tested on pigs with different levels of drug-evoked activity. An advanced pig tracking algorithm was developed in McFarlane and Schofield (1995) with special emphasis on the initial segmentation and background estimation. This approach is somewhat similar to ours, although we use the full RGB-space, another method for the position updates as well as a correction for the fish-eye lens.

In comparison to previous attempts, our system was developed to robustly track loose-housed pigs in real-world scenarios where main challenges are the pigs lying close together, uncontrolled background light and a large floor to be captured with the camera from a small height. The real-time implementation and test on ordinary PC technology illustrates that the method can be used in e.g. farm-management and automation systems or for behavior research in large loose-housed stables.

2. Pig tracking algorithm

The algorithm is a crucial part of any computer vision system and in this study, the main efforts have been in the algorithm development. The algorithm takes as input a time-sequence of color images (i.e. color video) and, after an initialisation procedure, it will estimate the spatial positions of each pig for each new frame.

Developing new algorithms always builds on existing methods. Our algorithm is most notably inspired by Wren (1996) and Wren et al. (1997), but several improvements have been made for the specific purpose of tracking pigs.

2.1. Basic tracking loop

The fundamental idea in the algorithm is to continuously update a so-called *support map* as well as a model for each pig. The support map for a specific pig is simply the coordinates of the pixels in a frame that are estimated to be part of the pig. As such, the support map can be formulated as a set of *xy*-coordinates in the image space. The model for each pig was chosen to be a Gaussian distribution in a 5D space which consists of the image *xy*-coordinates as well as RGB-color coordinates. As such, the 5D means and covariance matrices were estimated for each pig from the corresponding pixels in the support map.

When a new frame is received, the algorithm first updates the support map from the previous pig model and afterwards, using the new estimate of the support map, updates the pig model.

2.1.1. Updating the support maps

The support map of each pig is illustrated as the black regions on the pigs in Fig. 1. Several steps were taken to calculate these regions from the 5D Gaussian models.

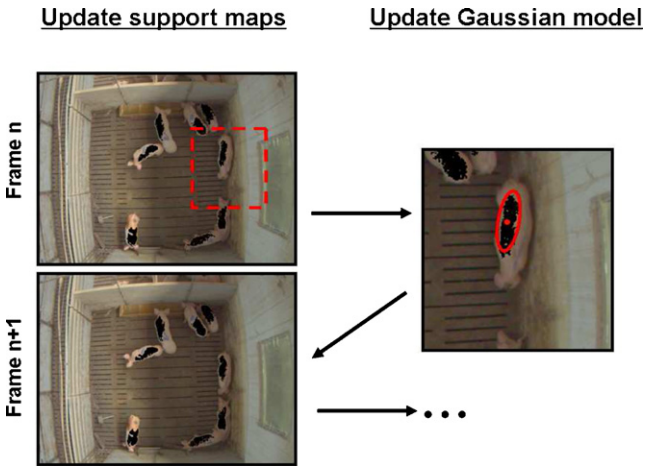

Fig. 1. Illustration of the basic flow of the algorithm. First, a new frame (frame *n*) is acquired and the support maps are updated from the previous Gaussian models. Each pig has its own support map which is shown as the black regions on the pigs. Next, the Gaussian models are updated from the support maps. Each pig also has its own Gaussian model. The mean is illustrated with a red dot and a contour line of the Gaussian distribution is shown as a red ellipse. Then, the process repeats when a new frame arrives (frame *n* + 1).

First, distance maps are calculated using the Mahalanobis distance from each pixel in the (new) image to the different pig models. One of the advantages of the 5D model is, that both the distance in RGB- and *xy*-coordinates are used and the Mahalanobis distance therefore both enforces spatial connectedness (a pig does not have any holes) as well as color similarity (most pigs have fairly uniform colors). The covariance matrices in the 5D model imply robustness towards variations in light intensity on the pig. Fig. 2 illustrates the spatial, color and combined 5D Mahalanobis distance map.

The next step is to find the pixels which are nearest (in Mahalanobis distance) to the pig models. A specific number of pixels are chosen for each support map since the areas of the pigs are fairly unchanged over time.

At this point, each pig model has an associated support map, but it will often contain pixels just outside the contour of the pig since these pixels are close to the *xy*-coordinate mean value of the pig model. One method to remove these erroneous regions of the support map would be to estimate the pig contours and use this information. However, as seen on Fig. 1, the pigs have a "natural" boundary of pixels around their perimeter with a different color from the center points of the pigs. This happens since the pigs are almost spherical of shape and, therefore, the light from above is not reflected back to the camera around the perimeter of the pig. Hence, the support map is split into several unconnected regions and the last step is to remove such unconnected regions from the support map. Our approach is simply to keep the largest connected region (blob).

It could be argued that the algorithm should consider more than one distance map at a time when calculating the support map for a given pig model. For instance, when two pigs are standing side-by-side and the mean value in *xy*-coordinates from the previous model is situated exactly between the two pigs, it will not be possible to determine whether the support map should be placed on one pig or the other (if similar colors are assumed). One method, that uses all distance maps at one time, would be to find the decision boundaries between the pig models as the minimum of all distance maps for a given pixel position. This method as well as variants of the method were tested, but the experiments showed that simply considering one distance map at a time gave better results.

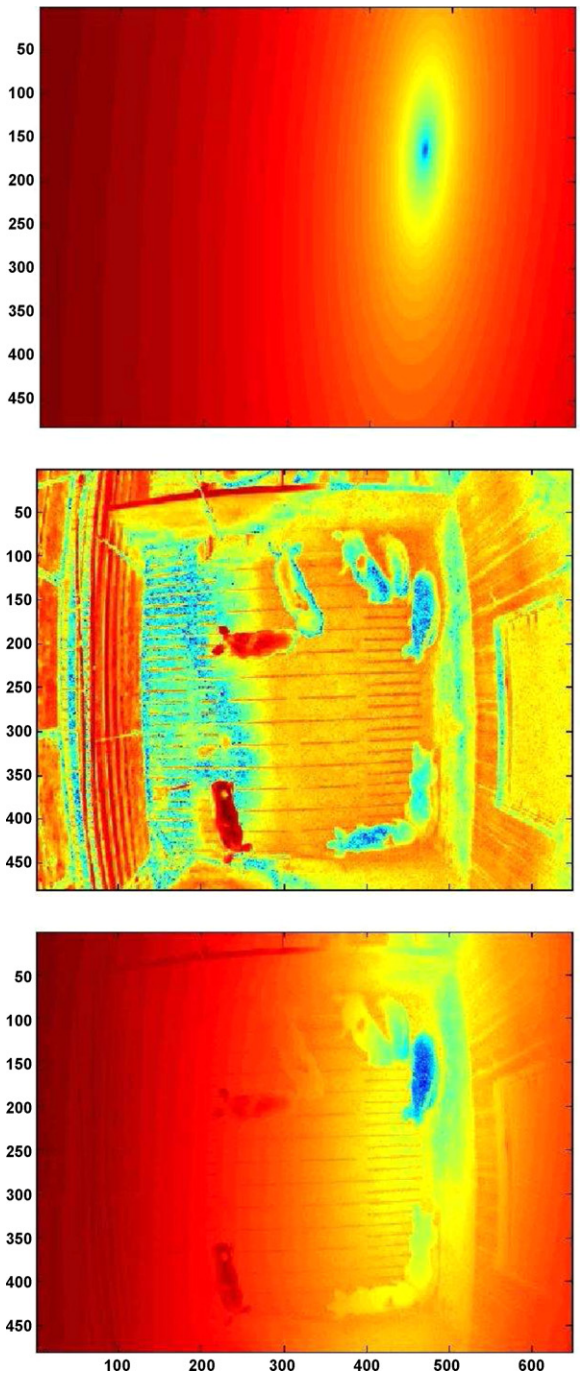

Fig. 2. Illustration of Mahalanobis distance maps between a specific pig model (with mean and covariance matrix) and all pixel positions in the image. The colors are scaled in the range from blue (shortest distance) to red (largest distance). The upper figure shows the spatial (*xy*-coordinates) distance map where the Gaussian shape is clearly seen. The middle figure shows the color (RGB-coordinates) distance map where it can be seen that several of the pigs have identical colors to the current pig (pig model). However, in the left part of the image, the pigs have other colors and the background is much closer to the current pig. This is due to light from a window in the right part of the image and the shadows that it creates. It illustrates some of the problems that can occur and the necessity of a combined 5D model with both position and color. The lower figure illustrates the combined 5D model distance map.

2.1.2. Updating the pig models

With the new support maps in hand, the pig models have to be updated. For each support map, the mean μ_{xy} and covariance matrix Σ_{xy} are calculated from the *xy*-coordinates of the pixels in the support map. Similarly, the mean μ_{rgb} and covariance matrix Σ_{rgb} are calculated from the RGB-values of the pixels. It can be assumed that *xy*-coordinates and RGB-space are not correlated and the updated parameters of the 5D model then becomes

$$\mu = \begin{bmatrix} \mu_{xy} \\ \mu_{rgb} \end{bmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} \Sigma_{xy} & 0 \\ 0 & \Sigma_{rgb} \end{bmatrix}$$

Pigs will often move in piece-wise straight lines over time which might suggest a Kalman-filtering approach to predict the value of μ_{xy} (and perhaps even Σ_{xy}) from previous observations. However, the experiments that were made did not show any advantages in that approach. Instead, the support map would often "wander away" from the pig when the pig changed direction. The experiments used a frame rate of 15 Hz and it is possible that the method would be more useful at higher frame rates since this would give less abrupt changes in the pigs movements.

Now, the pig models have been updated and the algorithm goes back to update the support map when the next image frame arrives.

2.2. Background/foreground estimation

To improve the performance of the algorithm, it is useful to estimate the background to avoid problems where the background has similar colors to the pigs. Additionally, the background is reasonably constant apart from variations in light intensity (e.g. due to roof-top windows). The method is to subtract the background estimate from the current image and perform a (fixed) thresholding to estimate the foreground (the pigs) as a mask. This mask is used in the update of the support maps in such a way that only pixels in the foreground estimate are allowed to become part of any support map.

The background is estimated by finding the mean value for every pixel position over the last few frames (10 frames in the case of a framerate of 15 Hz). Experiments were also made with medians and exponentially decaying running means, having approximately similar effect. The method adapts to many of the variations in light intensity that happens on short time-scales (order of few seconds) as limited by the number of frames in the circular image buffer. Still, the method is capable of removing the sudden movements of the pigs from the background as these movements are on an even shorter time-scale. Unfortunately the pigs often stay in one position for longer periods of time and will therefore become part of the background estimate. Our solution to this problem is to employ the support maps of the pigs. First, a dilation operation is performed on the support maps to make sure that also the boundary of the pigs are included and all the support maps are combined into one support map mask. Afterwards, the combined support map mask is used on the current image where the masked areas are replaced with the previous estimate of the background instead of the pig. Then, the image (without pigs) is added to the circular image buffer that is used to estimate the mean of the last 10 frames.

2.3. Initialisation

Before the main tracking loop (as described in Section 2.1) can begin, it is necessary to have an initial support map. Our approach has been to use a seed point in the center of the pig on the image and then use a square box of points around the seed as the initial support map. Afterwards, the pig model and support map are iteratively updated a few times on the same image to find a better estimate of the initial support map.

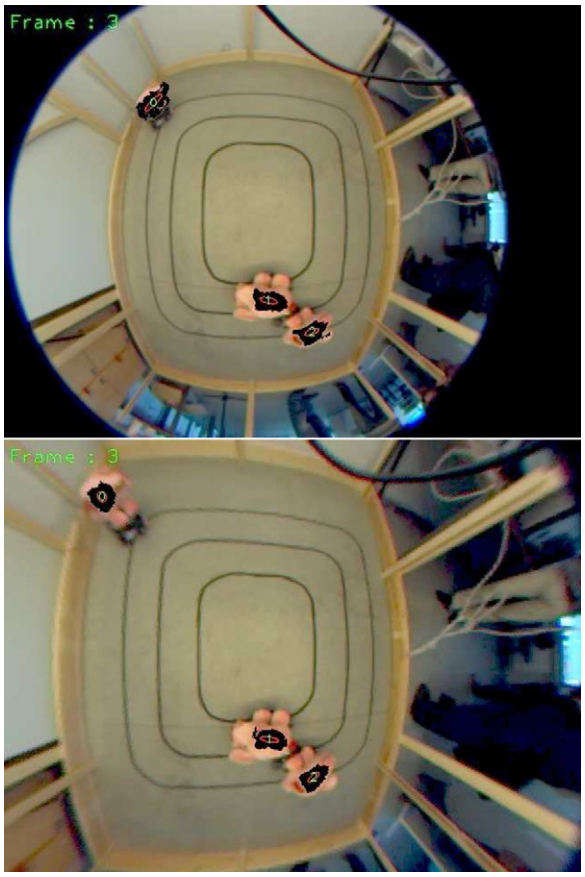

Fig. 3. Illustration of the view from the camera when a fisheye lens is used at the 3 toy pigs in the laboratory. The upper picture illustrates the camera view without fisheye lens compensation. The pig placed at the longest distance looks small and it is mostly seen from the side. The lower picture illustrates the camera view when the fisheye lens compensation is used. All 3 pigs have nearly the same size, but the sharpness of the pig placed at the longest distance is weak.

A variety of segmentation methods could have been used to estimate the initial support map or simply find the seed points. However, the algorithm was developed to be part of a larger automation system that would include subsystems such as automatic food dispensers where each pig would enter individually. Here, the pigs could be identified with certainty (e.g. with an RFID tag in the ear) and the initial seed point could be positioned on the pig in the same process. In the experiments, the user of the system simply uses the PC mouse to mark the seed point on the individual pigs in the initialisation phase.

3. Correction of fisheye lens distortion

Fisheye lenses were employed in some of the experiments, however, such lenses severely distort the image (as seen in Fig. 3). To avoid such distortion, we used a correction algorithm as described in Altera (2008). The basic idea is to map points from a fisheye projection (in this case, an equisolid angle projection) to a pinhole camera projection (rectilinear projection). This amounts to copying the RGB-values of pixel position (x_f, y_f) in the original fisheye image to a pixel position (x_p, y_p) in the rectilinear (corrected) image. The relation between (x_f, y_f) and (x_p, y_p) is given by

$$x_f = \frac{2x_p \sin(\tan^{-1}(\lambda/2))}{\lambda}$$

and

$$y_f = \frac{2y_p \sin(\tan^{-1}(\lambda/2))}{\lambda}$$

where

$$\lambda = \frac{\sqrt{x_p{}^2 + y_p{}^2}}{f}$$

and f is the focal length in pixels which can be calculated as

$$f = \frac{W}{4 \sin(FOV_{horz}/2)}$$

and FOV_{horz} is the field of view and W is the image width in pixels. The relation between (x_f, y_f) and (x_p, y_p) only has to be calculated once and is used as a look-up table. This is a computationally inexpensive method for correcting the fisheye lens distortion. Fig. 3 illustrates the effect of the correction algorithm.

4. Materials and methods

In the laboratory and stable tests a camera was mounted under the ceiling in a way that made it possible to cover the floor and the objects on the floor. The camera, an Elphel NC353L, was connected by an Ethernet cable to a PC. The PC was a Dell Optiplex 760 using an Intel Core 2 DUO CPU and 3.25 Gb RAM. The operating system was Windows XP, SP3. The standard lens was a Computar 4–8 mm, 1:1.4 and the fisheye lens was a Fujinon 1.4 mm, 1:1.4. The tracking software was modelled by using MatLab and the final implementation was done by transforming the MatLab code to C-code to achieve real-time performance.

In general, the time for each tracking experiment was set to a duration of 8 min. This duration was chosen from initial experiments which indicated that the tracking could (often) successfully be kept for at least 8 min, although this obviously varies in relation to the animal behavior.

4.1. Test in laboratory

Three pink toy pigs were used for the test. Each pig was mounted on the top of a small, programmable robot vehicle. The robot vehicle was equipped with a light sensor, and it was programmed to follow a black line at the floor with a constant speed, 0.16 m/s. The floor was illuminated by four 30 W fluorescent tubes. The lines on the floor were arranged as seen on Fig. 3. Each pig was following its unique route formed as a rectangle with rounded corners, and each pig was moving in opposite direction to the pig in the neighbour route. The distance between the routes was dimensioned so the pigs passed each other very closely to stress the camera vision algorithm. More precisely described, the test was performed as follows.

1. The first step of the test was to let the camera vision system track a toy pig, moving with constant speed in the inner rectangle.
2. After this, the camera vision system should track two pigs in the inner and the middle rectangle, moving towards each other with constant speed.
3. At last a third pig was placed in the outer rounded rectangle, moving with constant speed towards the pig in the middle rectangle.

The test should stress the camera vision algorithm and verify if it is possible to track 1, 2 and 3 pigs for 8 min.

The test was fulfilled for 3 different camera lens setups.

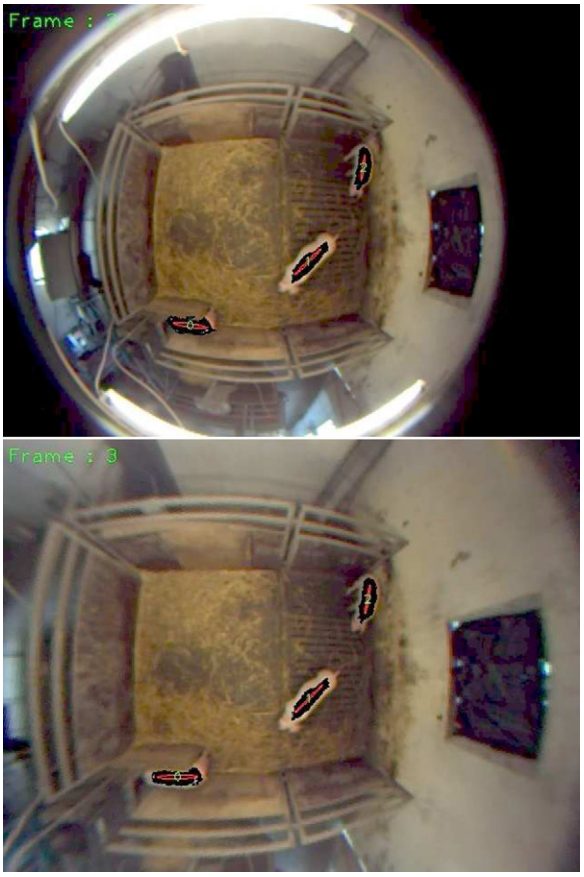

Fig. 4. Illustration of the view from the camera when a fisheye lens is used at the 3 real pigs in the stable. The upper picture illustrates the camera view without fisheye lens compensation. The pig placed at the longest distance looks small and it is mostly seen from the side. The lower picture illustrates the camera view when the fisheye lens compensation is used. All 3 pigs have nearly the same size, but the sharpness of the pig placed at the longest distance is weak.

A Standard lens that covered a small floor.
B Fisheye lens that covered a larger floor. No fisheye lens compensation software was used. The pig that was positioned most far from the fisheye lens was very small on the image.
C Fisheye lens that covered a larger floor. Fisheye lens compensation software was used this time.

The scenarios with 1, 2 and 3 toy pigs were repeated for the 3 lens setups described above (A, B and C), resulting in a total of 9 tests in the laboratory.

4.2. Test in stable

In the stable test a camera was mounted under the ceiling in a way so it could cover the floor including pigs at the floor. The floor was illuminated by two 30 W fluorescent tubes. 3 real, 10 weeks old pigs were used for the test. All pigs were healthy, very awake and curious. The test of the real pigs was performed similarly to the test in the laboratory. The camera vision system tracks 3 pigs simultaneously. The test should verify if it was possible to track all pigs for 8 min.

The camera vision stress scenarios, where 3 real pigs were moving for 8 min, was repeated for the 3 different lens setups (A, B and C). Therefore the total number of tests in the stable was 3 (Fig. 4).

5. Results and discussion

The laboratory test and the stable test gave the following results.

5.1. Results from laboratory

The results of the tracking quality test in the laboratory were as follows:

A Standard lens covered a small floor: scenarios with 1, 2 and 3 pigs could keep the tracking for 8 min.
B Fisheye lens covered a larger floor. No fisheye software compensation was used: scenarios with 1 and 2 pigs could keep the tracking for 8 min. In a scenario with 3 pigs it was not possible to track the pig that moved at the outer rectangle for more than 29 s.
C Fisheye lens covered a larger floor. Fisheye software compensation was used: scenarios with 1, 2 and 3 pigs could again keep the tracking for 8 min.

In all tests, the pig moving at the inner rectangle was added first. After this test the next pigs were added one by one, with an increasing distance from the standard lens.

When using the standard lens (test A), the size of the pigs did not differ much when the distance changed. The advantage of using a standard lens is that no correction algorithm is needed. The disadvantage is that the angle for the lens view is quite narrow. Therefore it is not possible to cover a circular area of the floor, which is much greater than the height to the ceiling.

By using a fisheye lens (tests B + C) it was only possible to track all pigs for 8 min when the correction algorithm was used for the camera tracking. If the correction algorithm was not used the pigs moving at the outer rectangles were very small, seen by the camera. By using the distance compensation algorithm the pigs that was far from the fisheye lens had nearly the same size as the size of the nearest pig, but the contour were more diffuse than the nearest pig. This loss of sharpness did not influence the tracking performance in this test.

5.2. Results from stable

The tracking test in the stable gave the following results.

A Standard lens covered a small floor: no scenarios with 3 moving pigs could keep the tracking for more than 17 s.
B Fisheye lens covered a larger floor. No fisheye software compensation were used: no scenarios with 3 moving pigs could keep the tracking for more than 34 s.
C Fisheye lens covered a larger floor. Fisheye software compensation were used: scenarios with 3 moving pigs could keep the tracking for 8 min.

When a standard lens was used (test A) it was not possible to track 3 pigs, because the coverage of the floor was too small, so the pigs moved immediately outside the picture area. When a fisheye lens was used without using the correction algorithm (test B), it was not possible to track the 3 pigs for more than 34 s. When a fisheye lens was used, combined with the correction algorithm (test C), it was possible to track the 3 pigs for 8 min.

5.3. Discussion

The system is still a prototype and several improvements could be made to increase the robustness such as a higher framerate

and better methods for background estimation. The tracking duration of 8 min is also somewhat arbitrary, but gives an idea about the general tracking performance of the system in a real-world scenario.

The tracking could be lost when the pigs were spinning rapidly on the floor or if they jumped. If the pigs climbed on top of each other it was not possible to keep the tracking of each pig.

It could be argued that the 3 pigs in our experiments are far from the loose housed systems with more than 50 pigs. However, we presume that our algorithm will be able to scale to larger numbers of pigs if more computing power is used to keep a sufficiently high framerate. The reason is that the same scenarios will occur for 3 pigs as for 50 pigs from the camera vision perspective. The most difficult problems occur when the pigs collide, overlap or are very close to each other and this situation is not much different for 3 or 50 pigs.

6. Conclusion

The described prototype system shows that it is possible to perform a real-time camera vision based tracking of loose-housed pigs in a real-world scenario. The system is able to keep the tracking of three live pigs for a duration of 8 min. It is shown that the tracking area can be increased by using a fisheye lens, but this requires a correction algorithm to be successful. The system is continuously able to point out the pigs at the floor. A standard PC was used for the tests in laboratory and stable. It will be possible to track more pigs by using a more powerful computer or specialized hardware.

Acknowledgement

The authors gratefully acknowledge Danish Agency for Science Technology and Innovation for providing financial support for this research.

References

Altera, 2008. A flexible architecture for fisheye correction in automotive rear-view cameras. Tech. rep., Altera Corporation.

Cangar, O., Leroy, T., Guarino, M., Vranken, E., Fallon, R., Lenehan, J., Mee, J., Berckmans, D., 2008. Automatic real-time monitoring of locomotion and posture behaviour of pregnant cows prior to calving using online image analysis. Computers and Electronics in Agriculture 64 (1), 53–60.

Damm, B., 2008. Loose housing of sows—is this good welfare? Acta Veterinaria Scandinavica 50 (Suppl. 1), 9.

Huhtala, A., 2007. Evaluation of instrumentation for cow positioning and tracking indoors. Biosystems Engineering 96 (3), 399–405.

Leroy, T., Vranken, E., Brecht, A.V., Struelens, E., Sonck, B., Berckmans, D., 2006. A computer vision method for on-line behavioral quantification of individually caged poultry. Transactions of the ASABE 49 (3), 795–802.

Lind, N.M., Vinther, M., Hemmingsen, R.P., Hansen, A.K., 2005. Validation of a digital video tracking system for recording pig locomotor behaviour. Journal of Neuroscience Methods 143 (2), 123–132.

McFarlane, N., Schofield, C., 1995. Segmentation and tracking of piglets in images. Machine Vision and Applications 8, 187–193.

Ni, L.M., Liu, Y., Lau, Y.C., 2004. Landmarc: Indoor location sensing using active RFID. Wireless Networks 10, 701–710.

Noldus, L.J., Spink, A.J., Tegelenbosch, R., 2001. Ethovision: a versatile video tracking system for automation of behavioral experiments. Behavior Research Methods, Instruments and Computers 33 (3), 398–414.

Noldus, L.P.J.J., Spink, A.J., Tegelenbosch, R.A.J., 2002. Computerised video tracking, movement analysis and behaviour recognition in insects. Computers and Electronics in Agriculture 35, 201–227.

Perner, P., 2001. Motion tracking of animals for behavior analysis. In: Proceedings of the 4th International Workshop on Visual Form (IWVF-4) , pp. 779–786.

Sergeant, D., Boyle, R., Forbes, M., 1998. Computer visual tracking of poultry. Computers and Electronics in Agriculture 21 (1), 1–18.

Shao, B., Xin, H., 2008. A real-time computer vision assessment and control of thermal comfort for group-housed pigs. Computers and Electronics in Agriculture 62 (1), 15–21.

Tan, K., Wasif, A., Tan, C., 2008. Objects tracking utilizing square grid RFID reader antenna network. Journal of Electromagnetic Waves and Applications 22, 27–38.

Tillett, R.D., Onyango, C.M., Marchant, J.A., 1997. Using model-based image processing to track animal movements. Computers and Electronics in Agriculture 17 (2), 249–261.

Wren, C., 1996. Pfinder: real-time tracking of the human body. Master's thesis, Massachusetts Institute of Technology (MIT).

Wren, C., Azarbayejani, A., Darrell, T., Pentland, A., 1997. Pfinder: real-time tracking of the human body. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 780–785.