Proceedings of the 2014 Federated Conference on Computer Science and Information Systems pp. 679–686
DOI: 10.15439/2014F195
ACSIS, Vol. 2

Indoor head detection and tracking on RGBD images

Katarzyna Niżałowska, Łukasz Burdka, Urszula Markowska-Kaczmar
Institute of Informatics
Wroclaw University of Technology
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
Email: urszula.markowska-kaczmar@pwr.wroc.pl

***Abstract*—A real-time human head detection and tracking method for a fall detection system is presented. It utilizes RGBD images to obtain a head position in the three-dimensional space. The proposed method is designed to be insensitive to a body orientation and requires no initial calibration for the tracked person. The evaluation was performed on the basis of annotated videos with realistic non-studio indoor everyday activities and falls. The proposed method outperforms head tracking from the Microsoft Kinect SDK skeleton tracking.**

I. Introduction

TRACKING of a human head is an important aspect of any system striving to monitor human behavior or health condition. Many conclusions can be made based on the information about a head position and orientation. A certain application that can benefit from a reliable information about a human head position is a fall detection for an elderly people monitoring system. Existing solutions focus mostly on detecting or tracking a human face instead of a head in general. Most of them also takes an assumption about constant vertical orientation of a human body. In a vast majority of situations such an approach is sufficient but in the context of a fall detection system there is no suitable existing solution.

The aim of our research was to develop a robust head detection and tracking method that is capable of tracking a human head regardless of its orientation and independently of a tracked person. The method uses joined color, motion and depth data to effectively perform this task. The introduced method maintain its performance in situations when the head position and orientation change rapidly such as during a fall.

The content of this paper is organized as follows. In section II related works in the field of head tracking are introduced. In section III we formulate the research problem, which our method is designed to solve. In section IV we describe the presented solution. Section V is dedicated to the evaluation of our method and contains the description of the experiment and the dataset followed by test results compared to the Kinect SDK head tracking [1]. In section VI we conclude our work and propose future work directions.

II. Related works

The head tracking problem has been widely studied over the past few years. In the literature, definitions of this problem describe different tasks. The majority of papers identifies the problem of head tracking with face tracking. They only tackle situations when the face is clearly visible on a video image and take the assumption that it is located near the camera, as in [2], [3], [4], [5], [6], [7]. Two most common applications of such defined head tracking are to obtain certain facial features [6], [2], [3], [4], and to approximate a spatial head orientation [5], [8]. In this paper the problem of head tracking refers to determining the position of a head regardless of its rotation around the vertical axis.

Since the information about a human head position and orientation can be utilized in a vast number of applications, there are many different approaches to solve this problem. In this paper we focus on vision systems as most versatile ones. The highest performance can be achieved using a thermal camera [9] as a data source. It is a consequence of a human head being easily distinguishable on thermal images. This solution, however, cannot be widely applied due to the high cost of thermal cameras. A common approach to this problem is using a video camera as a data source. The video camera was utilized in the methods described in [2], [3], [4], [10], [11], [6]. Recent appearance of affordable sensors containing both video and depth camera has exposed new possibilities in the field of image processing. A widely used device, integrating a depth sensor and a color camera is Microsoft Kinect. It is used for the head tracking task in [5] and [8]. Additionally Microsoft Company released SDK for Kinect [1], providing a skeleton tracking functionality. Thanks to this solution, if a skeleton is recognized properly by the Kinect sensor, information about a head position can be easily obtained, however, as shown in this paper, it lacks robustness.

Among vision systems utilizing different data sources, there are various methods solving the head detection problem. A method presented in [12] uses background subtraction to detect a moving silhouette and treats its highest point as a head. In [11] the background subtraction is also used to find interest points. Then, a classifier is applied. In [10] each tracked head must be initially introduced to the tracking system from four directions. In [3] and [7] only a face is detected using a generic Haar cascade face detector [13]. In this case, a face needs to be visible in satisfactory resolution.

After the head is detected, the tracking process can be initiated. Most methods assume an invariant orientation of a head during tracking. Therefore, a template is captured

978-83-60810-58-3/$25.00

once and subsequently SURF [7] or a Template Matching algorithm is used to track the template. Other methods use CAMSHIFT [2] algorithm or intensity gradient and color histograms analysis [10] to perform tracking.

Existing methods fail, when the head is seen from different perspectives or a human body is not arranged vertically, for example during a fall. They also focus only on using a single data source, while composition of color and depth information allows greater versatility of a tracking system.

III. Research problem

The method described in this paper is designed to solve the head detection and tracking problem with an assumption to apply it to an elderly people monitoring system. The detection task is defined as locating a single human head on an image regardless of its rotation around the vertical axis. The tracking task is interpreted as providing consecutive information about the location of the initially detected head in each frame. Given a streaming sequence of color and depth images obtained from a RGBD sensor, the method returns a spatial position of a tracked head or information that there is no head detected.

The returned position is defined in the 3-dimensional coordinate system described in section IV. The detection and tracking is performed in the real time, thus providing the position of a head in the last processed frame. The input stream is analyzed with the speed of fifteen frames per second. The following assumptions are made. The method tracks a single person in an indoor environment. Neither person-specific nor room-specific calibration needs to be performed. The method is sensor-independent, however, for a head to be detected, it should be located within the sensor depth range, which is from 0.8m to 4m for the Kinect. In order to perform tracking successfully, the initially detected head needs to be visible on a color image. It is not required to stay in the depth range. The method should be robust to rapid changes in a position and orientation of a head in situations such as a fall.

IV. Method description

Our method uses image processing techniques and simple decision rules. It utilizes a combined information, extracted from color and depth images, obtained from a RGBD sensor. The method consists of four modules. The interaction between them for the *n*-th frame is presented in the Fig. 1.

The first module is capable of creating a motion image, based on three consecutive color frames. The second module, referred to as Detector, detects a head based on current color, motion and depth frames. The Tracker module also uses current color, motion and depth frames to track the head, detected in the previous frame. In the fourth module, referred to as Integrator, the information about the head position provided by Detector and Tracker is integrated and the spatial position of the head is returned.

An interest point for the head detection is the top of every vertical silhouette, segmented from the depth image. The interest region is marked as a *head* if it is recognized as a face or a movement in this area is detected on the color image.

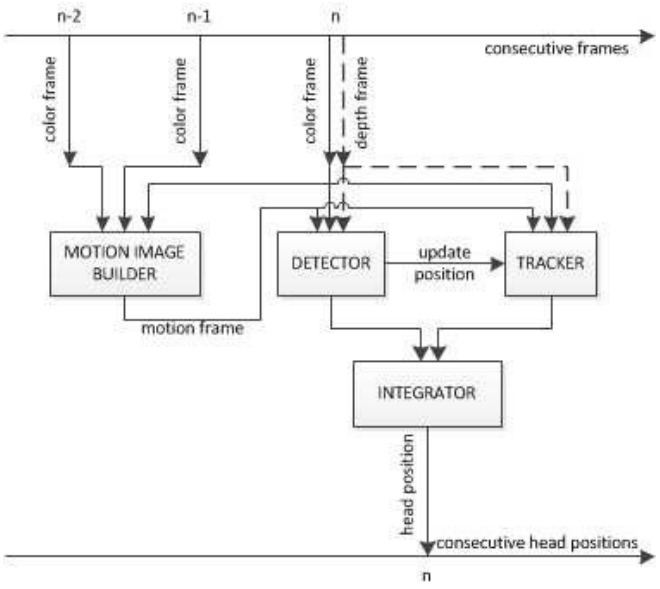

Fig. 1. The interaction between method modules for the *n*-th frame

TABLE I
Sensor-specific parameters

Symbol	Description	Kinect value
c_w, c_h	color image width and height [pixels]	640, 480
d_w, d_h	depth image width and height [pixels]	640, 480
sf	scaling factor	1.06
hd	horizontal displacement of the depth image on scaled color image [pixels]	6
hda	horizontal view angle of the depth camera [degrees]	58.5
vda	vertical view angle of the depth camera [degrees]	45.6
hca	horizontal view angle of the color camera [degrees]	62.0
vca	vertical view angle of the color camera [degrees]	48.6

The tracking is performed independently on the color image and on an image created as an absolute difference between consecutive color frames. This image will be referred to as a motion image. To increase the robustness of the tracking algorithm, both images are multiplied by the depth mask which cuts out the regions where the depth value differs significantly from the last known depth of the tracked head. The results returned by tracking on color and motion images are integrated as a weighted average based on the certainty of each of them.

A. Sensor-specific parameters

Due to a variety of RGBD sensors with different characteristics such as focal parameters, resolution and displacement of color and depth cameras, the presented method is parameterized to make it sensor-independent. Although the method was implemented and tested with the Kinect sensor, no additional Kinect SDK functionalities, such as skeleton tracking, were used to assure portability to other sensors. Defined parameters and their values for the Kinect sensor, described in the Kinect specification, are listed in the Tab. I

Color and depth image resolutions were selected from several options available for the Kinect sensor. The scaling factor was calculated with respect to the ratio of view angles of depth and color camera. The horizontal displacement is the consequence of displacement of cameras in the sensor. View angles are characteristics of the camera specified by a manufacturer.

To enable locating various points on the image in a real-world coordinate system, the information about the view angle of the depth camera and the depth value of the given point are used. The world coordinates are expressed in the right-handed coordinate system consistent with the one specified by Kinect SDK [1]. The origin is located at the center of the sensor, the Z-axis is pointing toward the direction of view and Y-axis points upwards. Whereas the image coordinate system has its origin in the top left corner of the image with the X-axis pointing to the right and the Y-axis pointing downwards. Image coordinate values are expressed in pixels. The real-world spatial coordinate (s_x, s_y, s_z) of a point (x, y) on the image, located at the distance d, measured in meters, is given by equations (1)

$$\begin{aligned} s_x &= 2d(\frac{x}{d_h} - 0.5)tg(\frac{hda}{2}), \\ s_y &= 2d(0.5 - \frac{y}{d_w})tg(\frac{vda}{2}), \\ s_z &= d, \end{aligned} \tag{1}$$

where d_w, d_h, hda, and vda are defined in the Tab. I.

The calculation allows a fast and accurate conversion from the depth image space to real-world coordinates. The choice of such coordinate system is justified by the possibility of comparing the method output with the Kinect skeleton tracking.

B. Preprocessing

The main problem, which needs to be solved in order to allow combining the depth and the color images is mapping between pixels of both images. The displacement of cameras and their different focal characteristics cause the difference between areas visible on both images. Kinect SDK provides an accurate conversion from the depth to color space. This conversion, however, only works in one direction and is only applicable to Kinect sensor. Additionally, it can only map a single depth pixel to color pixel which is computationally ineffective. A method similar to the Kinect solution was proposed in [14]. It is also unidirectional but in a contrast to the previous approach, it does not need Kinect sensor to be plugged in during its usage which allows wider application of this solution. It is only unidirectional since the precise inverse operation is not possible. It would require the analysis of depth pixels to find the one that matches best to the given color pixel. Such a solution would be very computationally complex. Since the existing methods do not use any fast bidirectional space mapping between color and depth images, a fast method of solving this problem is proposed. The color image is scaled and shifted to match the depth image. Given the resolution of the depth image d_w, d_h, the new size of the color image c_w', c_h' is calculated as in formula (2)

$$\begin{aligned} c'_w &= d_w sf, \\ c'_h &= d_h sf, \end{aligned} \tag{2}$$

where sf is the scaling factor defined in the Tab. I.

Subsequently, the color image is cropped to match the size of the depth image by extracting a sub-image of the size d_w, d_h, which top left corner is located in the point p_x, p_y calculated as in formula (3)

$$\begin{aligned} p_x &= \frac{c'_w - d_w}{2} + hd, \\ p_y &= \frac{c'_h - d_h}{2}, \end{aligned} \tag{3}$$

where d_w, d_h, and hd are defined in the Tab. I. The cropped subimage is presented in the Fig. 2.

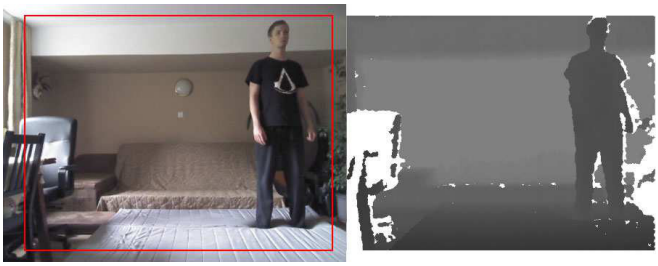

Fig. 2. Color image cropp rectangle

Once the transformation is completed, both images are aligned and no further calculations are necessary. The transformation is designed to give the best projection between images. While the method is not as accurate as the one in [14] or Kinect SDK mapping, it is bidirectional and much faster. The error of this method is greater for pixels located near the camera but it is tolerable at the range where the full body is visible.

The motion image is created based on the absolute difference between three consecutive color frames. It is a simplified version of a Motion History Image, described in [21]. A value of the pixel (x, y) is given by the equation (4)

$$\begin{aligned} M_n(x,y) = \ & |2(K_n(x,y) - K_{n-1}(x,y))| \\ & + |(K_{n-1}(x,y) - K_{n-2}(x,y))|, \end{aligned} \tag{4}$$

where $M_n(x, y)$ is the value of pixel (x, y) on the motion image, $K_{n-2}(x, y)$, $K_{n-1}(x, y)$ and $K_n(x, y)$ are values of corresponding pixels on three consecutive color frames. This approach is used to accurately highlight the area where the silhouette is currently located.

C. Head detection

The Detector module utilizes a current color, depth and motion frame and returns lists of *heads* and *uncertain heads* detected as a result for an input frame.

Algorithm 1 Head detection - finding regions of interest

```
image ← Dilatate(image);
image ← Canny(image);
contourList ← Group(image);
for all contour in contourList do
  d ← AvgDepth(contour);
  h ← Height(contour);
  if h ≥ hTreshold then
    rect ← CropV(contour, hHeight);
    median ← Median(rect, contour);
    if minHBreadth ≤ median ≤ maxHBreadth
    then
      regionOfInterest ← CropH(rect, median);
      interestRegions.Add(regionOfInterest);
    end if
  end if
end for
return interestRegions;
```

The head detection is performed in two steps. In the first one, regions of interest are found. An interest region is a sub-image that may contain a head. In the second step, each interest region is labeled either as a *head* or as an *uncertain head*.

Since our methods measures detected objects to rule out those, that cannot be human silhouettes, we define size constraints that need to be satisfied. Corresponding to the anthropometric studies of a human body [15], [16], the average breadth of a human head is 13.9 cm for men and 13.3 cm for women, while its length is 18.0 cm for men and 17.2 cm for women and its height is 21.2 cm for men and 19.8 cm for women. Additionally, according to [17], the height of a human body exceeds 1m after being four years old. Based on the introduced measurements and taking into account various transformations applied to the processed image, we define the following parameters:

- *hHeight* - the height of a head (26.4 cm),
- *minHBreadth* - the minimum width of a head (12 cm),
- *maxHBreadth* - the maximum width of a head (24 cm),
- *hTreshold* - the minimum height of a silhouette (1 m).

The proposed values exceed the top and bottom limit of a size of an adult human in order to avoid an omission of any real head detection.

During the first stage of the head detection, the depth image is analyzed. We use it to detect regions of interest. The aim of its analysis is to find silhouettes of a human-like shape. The process of finding interest regions is presented in the Algorithm 1.

Firstly, the image is repeatedly dilated to rule out the noise and erroneous pixels, which are white pixels, indicating no depth data. It is then binarized by Canny edge detector. Edge points are grouped into contours using a method described in [18]. As a consequence of the dilatation, the average size of a head on the image is increased by approximately 20%. This information is vital since in our method, boundaries for various measurements of a human body are defined in the real-world coordinate system. For each silhouette, its average depth is calculated to determine its distance from the camera. Having the distance, the height of the human silhouette is calculated using formulas (1). Only silhouettes that are at least the height of *hTreshold* are considered as possible human silhouettes, others are instantly eliminated. Subsequently, the bounding box of each silhouette is cropped leaving only its top *hHeight*. The median of the width of the silhouette's part located within a cropped rectangle is calculated. The rectangle is rejected if the calculated median is outside of the range from *minHBreadth* to *maxHBreadth*. Otherwise, the rectangle is cropped horizontally to obtain the interest region of a width equal to the introduced median.

Algorithm 2 Head detection - head classification

```
for all region in interestRegions do
  if FaceDetected(region) then
    heads.Add(region);
  else
    if MovementDetected(region) then
      heads.Add(region);
    else
      uncertainHeads.Add(region);
    end if
  end if
end for
```

After finding regions of interest, each of them is labeled as a *head* or an *uncertain head*. The classification process is shown in the Algorithm 2.

A simple decision sequence is used for this labeling task. Initially a Haar cascade is used to detect a face in the given region. We use the cascade model for frontal face recognition provided by OpenCV [19]. If the face is detected, the region is labeled as a *head*. Otherwise, the occurrence of the movement is checked in the interest region. If the movement is detected, the region is labeled as a *head*. Otherwise it is labeled as an *uncertain head*. To perform the movement detection, the motion image is thresholded to filter out the noise and the erosion is performed to clear the stronger noise. Subsequently the image is transformed by a multiple dilatation to highlight the movement region. The process is illustrated in the Fig. 3. If the region being an *uncertain head* contains at least 20% of white pixels we treat it as a *head*.

This approach has the following justification. The situation when there is a vertical silhouette detected, which contains an object of a size matching a human head in its top part, is not sufficient to classify this object as a head. However, because a human is not able to stand still without any movement, the movement is a reasonable indicator of a human presence.

D. Head tracking

The Tracker module takes a current color, depth and motion frame as an argument and, using the information about the previously detected head, performs tracking and returns up to

Fig. 3. Color image (left), motion image (center), and post-processed motion image (right)

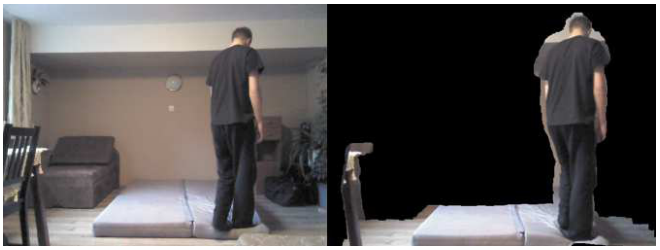

Fig. 4. Color image (left) and masked color image (right)

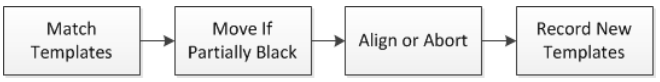

Fig. 5. Single tracking step for one image

4 possible head positions together with their *certainty factors*.

Tracking is initiated when an area labeled as a *head* is detected. It is performed independently on the color image and on the motion image introduced in section IV. Both images are multiplied by a binary mask cutting off areas where the depth value is significantly different from the depth value of the head. Such a mask is referred to as a depth mask.

In order to obtain the depth mask for each frame, auxiliary tracking is performed on the depth image. When the head is detected, a sub-image containing the head is recorded as a template. In the subsequent frame, an attempt is made to match the recorded template on the image. The search is performed in the region of interest, specified as the area containing the head in the previous frame, extended in each direction by a certain margin. The margin is designed to cover the distance that a head can traverse during the time of one frame. It is expressed in meters and transformed into pixels using the distance from the head to the sensor. The width of the margin is a parameter of the method and should be adjusted to match needs of the application. In our case, the video's frame rate is equal to 15 frames per second, thus the time span between consecutive frames is 66ms. For the fall detection system, where the speed of the head can be high, the reasonable margin value is 0.5m.

The template matching within the region of interest is performed using the Normalized Cross Correlation method (NCC), described in [20]. Once the best match is found, the template is updated with the newly found region, keeping its original size. The method returns a certainty measure to describe the quality of the match. This value is recorded together with a template position and will be referred to as a *certainty factor*. In the next step, the depth of the head is calculated and the depth mask is created as a binary image. The image has a black background and contains white pixels only in the regions, where the difference of the depth value and the depth of the head is not greater than the margin value.

Subsequently, the color and the motion images are multiplied by the depth mask. As a result of the multiplication, the areas where the depth value differs from the depth of the head are black, and those with a similar depth value are left unchanged. The example of a depth masking is shown in the Fig. 4.

Once masking is done, tracking on color and motion images is performed. For both images the same technique is used. The single step of tracking on one image is presented in Fig. 5. For each image, two templates are recorded. One having the size of the detected head and the other, located centrally within the first one and having a half of its size. It has been noticed that different sizes of the template perform better under different conditions, therefore tracking is executed independently for two templates. NCC is used to find the best template match. Next, the correction of the determined template location is done. In case a part of the template contains a large number of black pixels, its location is moved in the opposite direction.

If the best match of the template is located entirely in the black area, its position is aligned to the second tracked template. If this situation occurs to both templates simultaneously, tracking on the image is terminated.

Tracking is executed in parallel to the head detection process for each frame. If the *head* is detected within the range of the margin value of any of tracked templates, the size and position of each tracked template, including the auxiliary depth template, is updated with the area of the newly detected head. If no *head* is detected but there is an *uncertain head* in the rectangle overlapping one of the tracked templates, it is considered a *head* and all templates are updated as stated above.

After tracking is ceased, the positions of the tracked templates are returned together with their *certainty factors*.

E. Result integration

The Integrator module is capable of integrating the results returned by the Detector and the Tracker. The output of this module is a 3-dimensional position of a single head or information, that there was no head detected.

If any *head* was detected by the Detector, the results of the Tracker are ignored and the detected head is treated as a final result. Otherwise, the results of tracking are integrated in the following manner. In each frame there are two templates tracked on the color image and additional two on the motion image, therefore tracking results need to be integrated in order to provide a single location of the head. The location of the head is determined as a weighted average of all template positions. A *certainty factor* is used as a weight. This approach

Fig. 6. The example of a color frame (left) and a depth frame (right) from the dataset

causes inaccurate matches having less impact on the final outcome. Only templates matched on the color and the motion images are used in the integration process, because tracking on the depth image tends to be less accurate.

In order to obtain the spatial location of the tracked head, the formula (1) is used with the head position on the image and its depth value. It is then returned as a result of this module and the whole method.

V. Evaluation

The evaluation of the method was performed to assess its effectiveness and to compare it to the existing solution, implemented in the Kinect SDK. Since there is no RGBD benchmark dataset for the head tracking problem, the proper dataset was prepared. The dataset is described in details in the next subsection. Subsequently, the evaluation procedure, including experimental method and description of used measures, is presented. Finally, the results for our method and the Kinect SDK are shown and commented.

A. Dataset description

The dataset consists of 480 short films, recorded at 15 fps and containing video and depth images for each frame. All scenes are recorded in the indoor scenery and last from 6s to 24s, averagely 13s. Each scene shows one of two actors: a man or a woman. Each film presents either a daily action or a fall. The following actions were recorded: walking, standing, sitting on a chair, sitting on the ground, bending down, lying on a bed, lying on the ground, standing on a chair, cleaning, falling forward, falling backward and falling sideward.

Each action was recorded 20 times per actor and, excluding lying on a bed, contains records, where the actor was viewed from the front, back, left and right. Each film begins showing the empty room and captures the moment, when the person enters the frame and, except falls, finishes after the person leaves the frame. Films showing falls end when a person is lying on the ground. The example frame is shown in Fig. 6.

The dataset was annotated with a current head position to enable the evaluation of a head detection method, however, due to laboriousness of the frame annotation process, for each film, only 3 frames were annotated. They were located at 1/4, 1/2 and 3/4 of the film duration. The first and the last frame was not taken into account, because a vast majority of them was showing only the empty room. The annotation was performed

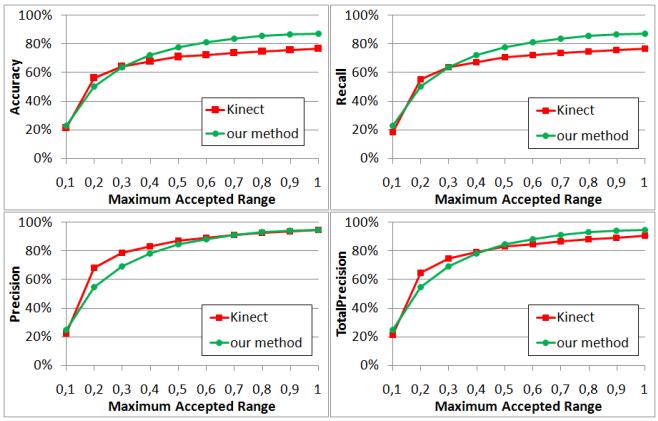

Fig. 7. Accuracy, Recall, Precision and TotalPrecision as a function of accepted range

by one person and validated by another one. On each annotated frame, the head position is specified or the frame was labeled as containing no head. The head position was labeled as a single point on the color image, located in the center of a head and transformed, using the depth information, into the real-world coordinate system. Together the 1440 frames are annotated in 480 films. In 1350 of them, there was a head visible in the frame, and in 90 of them, there was no head visible.

To compare our method to the method from the Kinect SDK, the skeleton information obtained from the Kinect sensor is also recorded for each frame.

B. Test method

Both methods were evaluated using various measures to provide a more comprehensive analysis of their performance. They were then compared to recognize differences in their functioning.

In the first step of the evaluation process, each film was replayed and analyzed by our method to detect and track the head in the real time. In each film, 3 frames, that are annotated with the head position, were evaluated. The number of heads annotated, tracked by our method and tracked by the Kinect SDK was stored for each frame. Additionally, if both numbers of annotated and tracked heads were greater or equal to 1, the following distances were calculated:

- the 3-dimensional Euclidean distance between tracked and annotated head, measured in meters, presented in formula (5),
- the distance between vertical components of positions of both heads, measured in meters, calculated as in (6).

$$d = \sqrt{(t_x - a_x)^2 + (t_y - a_y)^2 + (t_z - a_z)^2}, \quad (5)$$

$$d_v = \sqrt{(t_y - a_y)^2}, \quad (6)$$

where d is the 3-dimensional distance, (t_x, t_y, t_z) is the position of a tracked head in the real-world coordinate system, (a_x, a_y, a_z) is the position of an annotated head in the real-world coordinate system and d_v is a distance between vertical components of those points.

In order to consider different precision requirements, various measures were calculated as functions of the maximal distance between the annotated and the tracked head to treat it as detected correctly. The maximal distance is referred to as a range and extends from 10 cm to 1 m with 10 cm intervals.

The measures presented in the formula (7) were calculated for both our method and the Kinect SDK method

$$\begin{aligned} Accuracy(r) &= \frac{TP_r + TN}{AF}, \\ Recall(r) &= \frac{TP_r}{AH}, \\ Precision(r) &= \frac{TP_r}{DH}, \\ TotalPrecision(r) &= \frac{ATP_r}{ADH}, \end{aligned} \tag{7}$$

where: r is the range, the measure is calculated for, TP_r is a number of frames, in which the head was detected within a range r, TN is the number of frames, correctly classified as containing no head, AF is the number of annotated frames (1440), AH is the number of frames, annotated as containing a head (1350), DH is thenumber of frames, in which at least one head was detected, ATP_r is the number of heads, detected within a range r, including multiple heads detected in one frame, and ADH is the number of all detected heads, including multiple heads detected in one frame.

Due to the fact, that our method is designed to track one head for each frame and Kinect SDK can track higher number of heads, to calculate *Accuracy*, *Recall* and *Precision* we choose one head tracked by the Kinect, closest to the head annotated in this frame, and compare it to the result of our method. Only while calculating the *TotalPrecision*, we take into account all the heads tracked by the Kinect independently, even if there were more than one head in a given frame.

C. Experimental results

In this subsection, results of the experiment described previously are presented and commented. Fig. 7. shows the comparison of *Accuracy*, *Recall*, *Precision* and *TotalPrecision* calculated for our method and for the Kinect SDK in the function of the range.

As it can be seen, the *Accuracy* and *Recall* of our method is mostly higher than of Kinect, except for the range of 20 cm while the *Precision* of Kinect is greater than ours considering the range from 20 cm to 60 cm. For the rest of ranges, *Precision* of both methods is comparable. However, the *TotalPrecision* calculated for all heads detected by the Kinect is only greater in the range of 20 cm to 40 cm, while in the range wider than 50 cm, our method outperforms the Kinect. It is also notable, that considering the range lower than 10 cm, every measure is slightly higher for our method. This can be caused by the fact, that Kinect tends to detect the head on the chin level or even on the neck, while the dataset was annotated with points located in the center of the head, approximately on the line of ears and eyes. The significant difference between the *Precision* and *TotalPrecision* of Kinect indicates the high number of heads detected wrong while there was another head detected correctly. It is important to highlight that in situations where the Kinect detected more than one head, only the best one was used to calculate *Accuracy*, *Precision* and *Recall* while the rest was ignored. The higher *Accuracy* and *Recall* of our method indicate that Kinect detects less heads but when it does, it tends to be more precise.

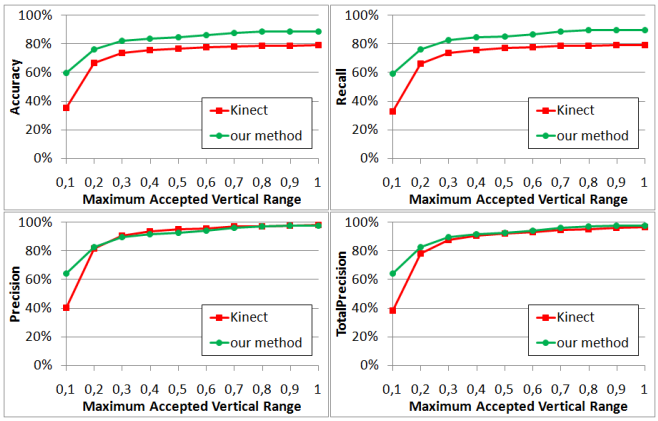

Fig. 8. Accuracy, Recall, Precision and TotalPrecision as a function of accepted vertical range

Regarding the fact, that our method is designed to match needs of a fall detection system, the correct vertical coordinate of a detected head is the one crucial for the proper functioning of such system. To asses both methods in terms of application in a fall detection system, the measures described by formulas (7) were calculated considering only the vertical component of the distance between the annotated and the tracked head. Results are presented in Fig. 8.

When only the distance between Y-coordinates is taken into account, *Accuracy* and *Recall* of our method is significantly greater than *Accuracy* and *Recall* of the Kinect. The *Precision* and *TotalPrecision* are also much greater considering the range of 10 cm and comparable for the rest of ranges.

VI. Conclusions

The above results lead to the conclusion that our method is more suitable for the application in a fall detection system than the method from the Kinect SDK. Among other possible applications, our method can be recommended for those, where the *Accuracy* and *Recall* are more important than the high *Precision*.

A. Summary

The method of human head detection and tracking on RGBD images was presented. As the evaluation shows, it outperforms the Kinect SDK skeleton tracking. Furthermore the method is independent of a used sensor and therefore its usage is not limited to the Microsoft Kinect. Promising

evaluation results indicate that it is a valuable head tracking method that can be successfully applied to tracking a single person in the indoor conditions. Our solution is particularly useful when there is no annotated dataset that could be used for any other machine learning approach since it requires no initial training and can be used to track any person without the necessity of previous calibration. Furthermore it is suitable for the fall detection task as it maintains its effectiveness during a fall. Therefore it can be used either as a primary data source for a newly developed fall detection system or as an auxiliary tracking method to boost the robustness of an existing fall detection system.

B. Future works

During the development and evaluation of our method, we identified various improvements that could potentially increase its effectiveness. During the classification of interest regions recognized as *uncertain heads*, a machine learning approach can be used to decide if the region should be classified as a *head*. Such a solution would require annotated objects, which are not heads, as negative examples and use them together with positively annotated heads to train a classifier. For that approach to be effective, the size of the dataset should be greater than the one used during the development of our method. The necessity to train a classifier would decrease the assumed versatility of our method but could improve its effectiveness in a specific target scenario. Potentially useful features for the classification task would be shape descriptors and color histogram components.

Another promising improvement is the dynamic adjustment of the margin value used during the tracking process to define the area where the search for a tracked head is performed. At present, the size of the margin is fixed to a reasonable value of 0.5m, which is suitable for the fall detection task. This value however could be adjusted based on the current speed of the tracked head.

Our method was designed to solve the head detection and tracking problem for a single person in the room, however it can easily be extended to track any number of heads. Such a modification would require developing a method of matching tracked and detected heads to allow integrating positions of corresponding heads.

Even though the presented configuration of our method has proven to be effective, its further improvements are still a subject of research. Since the method is flexible and divided into modules, each module can be used, modified and adjusted to fit special requirements independently.

REFERENCES

[1] Microsoft. Kinect SDK reference. [Online]. Available: http://msdn.microsoft.com/en-us/library/hh855347.aspx

[2] G. Bradski, "Real time face and object tracking as a component of a perceptual user interface," in *Proceedings Fourth IEEE Workshop on Applications of Computer Vision. WACV '98*, Oct 1998, pp. 214–219. [Online]. Available: http://dx.doi.org/10.1109/ACV.1998.732882

[3] Y. Li, H. Ai, C. Huang, and S. Lao, "Robust head tracking based on a multi-state particle filter," in *7th International Conference on Automatic Face and Gesture Recognition. FGR 2006*, April 2006, pp. 335–340. [Online]. Available: http://dx.doi.org/10.1109/FGR.2006.96

[4] P. Fieguth and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates," in *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun 1997. ISSN 1063-6919 pp. 21–27. [Online]. Available: http://dx.doi.org/10.1109/CVPR.1997.609292

[5] S. Li, K. N. Ngan, and L. Sheng, "A head pose tracking system using RGB-D camera," in *Proceedings of the 9th International Conference on Computer Vision Systems*, ser. ICVS'13. Berlin, Heidelberg: Springer-Verlag, 2013. ISBN 978-3-642-39401-0 pp. 153–162. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-39402-7_16

[6] M. Weber, W. Einhauser, M. Welling, and P. Perona, "Viewpoint-invariant learning and detection of human heads," in *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, 2000, pp. 20–27. [Online]. Available: http://dx.doi.org/10.1109/AFGR.2000.840607

[7] J. Choi, Y. Dumortier, S.-I. Choi, M. Ahmad, and G. Medioni, "Real-time 3-D face tracking and modeling from awebcam," in *IEEE Workshop on Applications of Computer Vision (WACV)*, Jan 2012. ISSN 1550-5790 pp. 33–40. [Online]. Available: http://dx.doi.org/10.1109/WACV.2012.6163031

[8] F. Kondori, S. Yousefi, H. Li, S. Sonning, and S. Sonning, "3D head pose estimation using the kinect," in *International Conference on Wireless Communications and Signal Processing (WCSP)*, Nov 2011, pp. 1–4. [Online]. Available: http://dx.doi.org/10.1109/WCSP.2011.6096866

[9] W. K. Wong, Z. Y. Chew, C. K. Loo, and W. S. Lim, "An effective trespasser detection system using thermal camera," in *Second International Conference on Computer Research and Development*, May 2010, pp. 702–706. [Online]. Available: http://dx.doi.org/10.1109/ICCRD.2010.161

[10] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun 1998. ISSN 1063-6919 pp. 232–237. [Online]. Available: http://dx.doi.org/10.1109/CVPR.1998.698614

[11] V. Subburaman, A. Descamps, and C. Carincotte, "Counting people in the crowd using a generic head detector," in *IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, Sept 2012, pp. 470–475. [Online]. Available: http://dx.doi.org/10.1109/AVSS.2012.87

[12] M. Munaro, F. Basso, and E. Menegatti, "Tracking people within groups with RGB-D data," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2012. ISSN 2153-0858 pp. 2101–2107. [Online]. Available: http://dx.doi.org/10.1109/IROS.2012.6385772

[13] P. I. Wilson and J. Fernandez, "Facial feature detection using Haar classifiers," *J. Comput. Sci. Coll.*, vol. 21, no. 4, pp. 127–133, Apr. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1127389.1127416

[14] N. Burrus. (2011, Aug.) Kinect calibration. [Online]. Available: http://burrus.name/index.php/Research/KinectCalibration

[15] A. Poston, *Static adult human physical characteristics of the adult head*. Washington DC, USA: Department of Defense Human Factors Engineering Technical Advisory Group, 2000. [Online]. Available: http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA467401

[16] Z. Zhuang and B. Bradtmiller, "Head-and-face anthropometric survey of U.S. respirator users," in *Journal of Occupational and Environmental Hygiene*, vol. 2. ACGIH, 2005, pp. 567–576. [Online]. Available: http://www.nap.edu/html/11815/Anthrotech_report.pdf

[17] C. for Disease Control and Prevention, *Anthropometric reference data for children and adults: United States, 2003-2006*. USA: National Center for Health Statistics US Department of Health and Human Services, 2008. [Online]. Available: http://www.cdc.gov/nchs/data/nhsr/nhsr010.pdf

[18] S. Suzuki and K. be, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32 – 46, 1985. [Online]. Available: http://dx.doi.org/10.1016/0734-189X(85)90016-7

[19] A. F. Reimondo. Haar cascades. [Online]. Available: http://alereimondo.no-ip.org/OpenCV/34/

[20] K. Briechle and U. D. Hanebeck, "Template matching using fast normalized cross correlation," in *Proc. SPIE*, vol. 4387, 2001, pp. 95–102. [Online]. Available: http://dx.doi.org/10.1117/12.421129

[21] A. Bobick and J. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257–267, Mar 2001. [Online]. Available: http://dx.doi.org/10.1109/34.910878