in: IEEE Int. Symp. on Robot and Human Interactive Communication (RO-MAN), Lisbon, Portugal, pp. 74-80, IEEE 2017

I See You Lying on the Ground - Can I Help You? Fast Fallen Person Detection in 3D with a Mobile Robot

Benjamin Lewandowski, Tim Wengefeld, Thomas Schmiedel and Horst-Michael Gross

***Abstract*— One important function in assistive robotics for home applications is the detection of emergency cases, like falls. In this paper, we present a new detection system which can run on a mobile robot to detect persons after a fall event robustly. The system is based on 3D Normal Distributions Transform (NDT) maps on which a powerful segmentation is applied. Segments most likely belonging to a person lying on the ground are grouped into clusters. After extracting features with a soft encoding approach, each cluster is classified separately. Our experiments show that the system is able to reliably detect fallen persons in real-time. It clearly outperforms other 3D state-of-the-art approaches. We can show that our system is able to handle even very challenging situations, where fallen persons are very close to other objects in the apartment. Such complex fall events often occur in real-world applications.**

I. Introduction

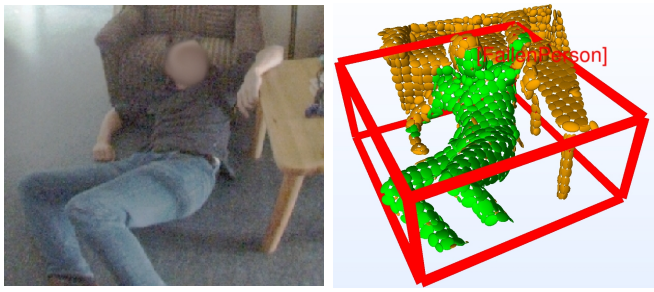
Having an independent life in ones own apartment as long as possible is a big desire for many people. Unfortunately, mobility and fitness decrease with proceeding age and, thus, a lot of ordinary activities can become challenging. Simultaneously, the risk for dangerous situations increases which is especially critical when living alone.

One example for such critical situations are falls. According to [1], about 30% of the population at the age of 65 and above fall at least once a year. Consequences of a fall event reach from smaller injuries, like contusions, up to more critical injuries, like fractures. The higher the age, the higher is the risk for such an injury. Besides the physical impairments, a fall can also have psychological consequences, e.g. the fear of falling again. This can cause social retreat and decreasing activity. Hence, a fall event is one of the main reasons for people to decide to live in a supervised retirement home.

Nursing staff and caregivers can assist the elderly in their everyday life and can check if they are fine in their own home. But due to the demographic change in industrialized countries the support by nursing staff gets more and more difficult. According to [2] people at the age of 65 and higher will represent more than 30% of the German population in the year 2060, while the working population will decrease by about 23% compared to the year 2013. A way to unburden nursing staff is the commitment of a robot who lives together with people or supports them elsewhere, like e.g. explored in [3]. Amongst other things, a robot could keep an eye on a person in certain time intervals. Critical situations can then be reported to human caregivers over a communication system, or the emergency medical service could be called directly. Besides the detection and report of dangerous situations, the pure presence of a robotic companion could also lower the fear of such dangers.

All Authors are with Neuroinformatics and Cognitive Robotics Lab, Technische Universität Ilmenau, 98694 Ilmenau, Germany. benjamin.lewandowski@tu-ilmenau.de

This work has received funding from the German Federal Ministry of Education and Research (BMBF) to the project SYMPARTNER (grant agreement no. 16SV7218), and by the German Federal Ministry of Education and Research (BMBF) to the project 3D-PersA (grant agreement no. 03ZZ0408) in the program Zwanzig20 – Partnership for Innovation as part of the research alliance 3Dsensation.



(a) (b)

Fig. 1: Difficult example for a fallen person. The RGB image in (a) shows that the person touches a table and an armchair. Nevertheless, our detector is able to detect the person as displayed in (b).

In this work, we present a novel 3D lying person detector for detecting falls which is applied on our mobile companion robot [3]. This multi-stage approach is based on a robust segmentation of a 3D Normal Distributions Transform (NDT) map [4]. The goal of the segmentation is to find clusters of NDT cells which probably correspond to a fallen person. From each cluster features are extracted using a soft encoding approach. Finally, these features are used to classify whether a given cluster represents a fallen person or not. As shown in Fig. 1 our approach is able to detect fallen persons even in difficult situations, e.g. when they touch furniture, like armchairs or tables. For cluster-based approaches such scenes are very difficult because objects included within a cluster could cause a different feature representation and, thus, a correct classification becomes more complicated.

II. Related Work

The literature about fall detection systems is comprehensive, especially in the domain of wearable devices. Overviews about existing approaches are given in [5], [6], [7] and [8]. Most of the existing systems try to detect a fall in the moment it is happening, i.e. the actual movement of the human body until it hits the ground. Many detectors rely on wearable devices, like wristbands or belts, which measure the acceleration. Modern variants, like [9] and [10], can utilize

the accelerometers of a smartphone and thus, no extra device must be worn. Nevertheless, a fall cannot be detected when people forget their mobile phone somewhere. Furthermore, the deployment of wearable sensors on a human is inappropriate for the operation of a mobile robot since the detection should be independent of external devices.

Also vision based systems exist which analyze the velocity or form of a person detected and tracked over time. Recent approaches also exploit the skeletons delivered by a 3D sensor for fall detection, e.g. the work in [11] or [12]. But for our scenario an active detection via vision-based sensors is unsuitable because this would require that the robot follows the human permanently without a break and thus, a very intrusive movement behavior would be necessary.

Hence, we define the problem of fall detection for our scenario as the detection of people lying on the ground. This offers multiple advantages: The robot does not have to observe a fall when it happens. The fall can rather be detected afterwards. In the same sense, the robot has several chances to detect the fall from different points of view when going on a search mission. Furthermore, the fall detector does not need to run all the time. It only has to be enabled in cases where the human cannot be detected by any of the robot's regular person perception modules. Hence, the robot's on-board computation capacity can be used for other services and daily routines.

Unfortunately, literature about systems which detect persons lying on the ground via vision sensors is very rare. In [13] a part-based approach for the detection of upright pedestrians in 2D RGB images based on edge features [14] was adapted to lying persons. To handle the multitude of different lying poses, eight different detectors were trained – each for a certain orientation of the body. Similar, in [15] multiple SVM classifiers were trained. Disadvantages of these approaches are the run time and the number of required training samples. Each detector costs computation time and requires lying person samples of a particular orientation. Furthermore, RGB cameras are sensitive to lighting conditions and, thus, they cannot be used in the dark. In contrast, our approach does not require the training of multiple detectors. Furthermore, we make use of active 3D sensors which allow the application in darkness.

An 3D approach based on point clouds is presented in [16]. There, the first step is the removal of the ground plane followed by a Euclidean clustering. Then after extracting features based on surface normals each cluster is classified separately. The most problematic part of this approach is the clustering step. Since a ground plane subtraction is the only step of pre-segmentation, it can happen that a person is split into multiple clusters depending on the viewpoint of the sensor, or a human is represented by a cluster together with objects of the environment, e.g. wheeled walkers or armchairs. Several clusters of a single person divide the human's unique features and thus, lower their distinctiveness. This in turn could lead to a missing detection. In a similar way objects deform the feature representation of humans when included into a cluster.

Our system is a multi-stage approach too. But in contrast, we use NDT maps as 3D data representation instead of raw point clouds which allows an efficient fusion of sensor readings from different points of view. This should avoid that the representation of a person is split into multiple clusters. Furthermore, before the clustering step we do a more complex segmentation and thus, only the parts of the map which most likely belong to a lying person are processed further. Hence, the resulting clusters should be free from environment objects, even in cases where they are touched directly as shown in Fig. 1. In our experiments, we show that our approach clearly outperforms our earlier results [16].

III. NDT FALLEN PERSON DETECTOR

The architecture of our fallen person detector is shown in Fig. 2. The system consists of different modules which are described more detailed in the following. Input is a depth image captured by a depth sensor like the Asus Xtion or the Kinect2. After creating an NDT map of the scene appropriate features are extracted. These features are used for a segmentation step, where those NDT cells which most likely belong to a fallen person are searched and grouped into clusters. Afterwards, per cluster a single feature vector is calculated and used to classify whether a given cluster represents a fallen person or not.

A. Ground plane estimation

Since the ground of a scene does not hold relevant information which are useful for the detection of lying people, sensor readings belonging to the ground plane can be ignored. This procedure speeds up the whole detection process and simplifies the following segmentation process.

For the ground plane estimation different possibilities exist. For example the complete depth image or only a certain region could be converted into a point cloud on which a RANSAC algorithm [18] is applied in order to calculate the plane. Alternatively, the plane could be removed directly within the input depth image by the method of [19], where in a so called v-disparity image the depth values per horizontal image line are accumulated. Assuming that most of the points belong to the ground, the plane can be found by a simple line fitting within the v-disparity image. In cases where the extrinsic parameters of the depth sensor are known exactly, a costly calculation of the ground plane is not necessary because the plane is defined implicitly by the sensor mounting. When applying the detector on a mobile robot, like in our scenario, the latter approach should be the most appropriate choice since nearly no computing time is consumed.

B. NDT mapping

Our fallen person detector operates on NDT maps, a spatial representation storing surface parts as three dimensional normal distributions $\mathcal{N}(\vec{\mu}, \Sigma)$. Such a map is created by converting a depth image into a point cloud, which is superimposed by a set of voxels afterwards. Per voxel a mean vector $\vec{\mu}$ and a covariance matrix Σ can be calculated

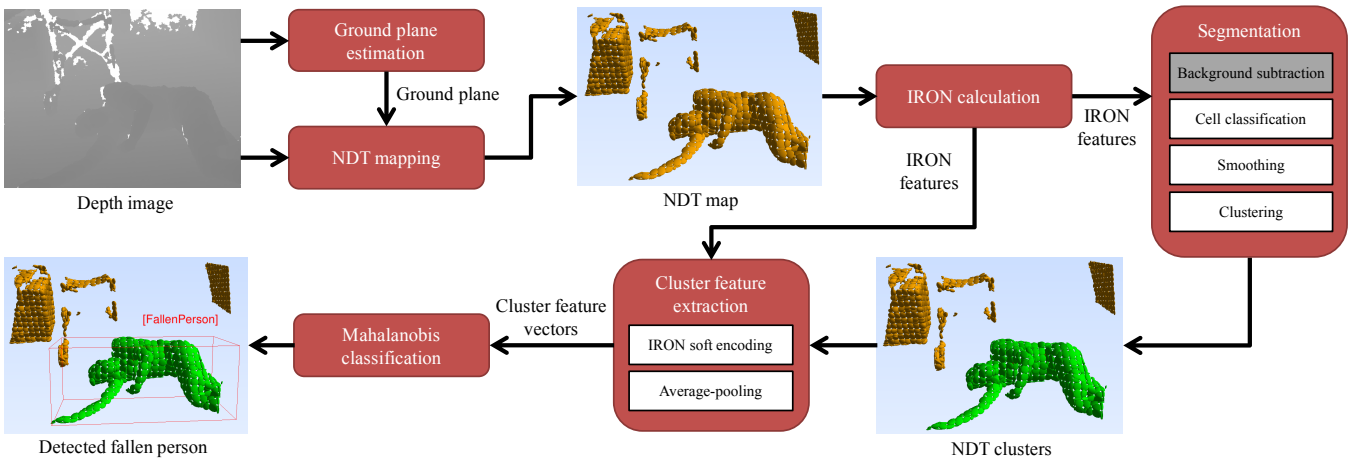


Fig. 2: Overview of the presented fall detection system. After computing a ground plane-free NDT map of the input depth image, IRON features [17] are calculated. Using these features, a segmentation on the map is applied in order to find clusters of NDT cells. Finally, each cluster is classified by a Mahalanobis classifier separately after extracting appropriate features per cluster.

out of the points [4]. The final map consists of a set of N normal distributions only, where N is the number of voxels which surround a certain minimum number of points. Please note, that we use the word "NDT cell" in the following as an equivalent for a single normal distribution. In 3D, these distributions can be visualized by ellipsoids as shown in Fig. 1b. In particular, we use the NDT mapping approach presented in [20] which accumulates sensor measurements over time and thus, creates a more complete representation. As voxel size we chose $5cm \times 5cm \times 5cm$.

The reason for using the NDT representation in this work are twofold. On the one hand, the subsequent calculations are much faster, compared to applying them on the raw point cloud, because the number of elements which need to be considered is much lower. On the other hand, our robot already uses an NDT map for localization and navigation and thus, no extra representation must be computed or stored. Since the ground plane of the scene is known in our system, the resulting map is free from NDT cells which belong to the floor as shown in Fig. 2.

C. IRON feature calculation

In order to detect fallen persons and to segment the scene, appropriate features have to be computed from the NDT map. The IRON descriptors [17] are utilized for this purpose. Originally, they were developed for the application in mapping and localization. However, we found out that IRON features are well suitable for a number of detection tasks, too. The reason for that is their rotational and translational invariance, which makes them very distinctive for persons in various poses. Simultaneously, their calculation does not need much computational resources.

Calculating IRON features requires a surface normal for each NDT cell. Such normal vectors are computed via an eigenvalue decomposition of the covariance matrices Σ of the NDT cells. There, the eigenvector corresponding to the smallest eigenvalue equates to a surface normal. The IRON-descriptor of a cell itself considers all neighboring cells within a spherical neighborhood (we chose a radius of $50cm$) and consists of two equally sized 2D histograms. Both histograms discretize the distance between the base cell and the neighboring cells in their first dimension respectively. The first histogram additionally encodes the angle between the base's normal and the normals of the neighboring cells. Hence, the first histogram corresponds to a description of the local surface curvature. The second histogram encodes the shape of the local neighborhood by binning the angle between the base's surface normal and the line connecting the means of the base and neighboring cells in the second dimension. In our experiments, we used three distance bins and eight angular bins for both histograms.

D. Segmentation

The final classification step of our detector evaluates clusters of NDT cells. Therefore, the goal of the segmentation step is to find all cells which most likely belong to persons lying on the ground and to combine them into groups. The inclusion of non-human objects or background elements, like furniture, would change the shape of a cluster. This complicates the detection because it gets more difficult to distinguish between clusters with and without persons. Hence, the idea behind the segmentation process is to filter out those non-human elements. The sub-steps to obtain such clusters are presented in the following.

1) Background subtraction: A simple idea to remove content belonging to the background is to employ a map of the environment if available. Since our robot uses such a map for localization, we can subtract parts of the background in our local map gathered by current sensor readings.

In particular, we have implemented a simple nearest neighbor

search, where for each NDT cell in the local map the closest cell in the environment map is searched. If both cells have a distance below a threshold τ to each other, it is assumed that they represent the same background element.
Since our approach is based on NDT cells, we do not need to rely on distances only. Instead, we additionally exploit the covariances to decide if a cell matches the background. More specifically, we include the similarity between observed and known background cells and, therefore, create a dynamic threshold τ per cell. To do so, we exploit the normal vectors of the cells. The smaller the angle between the normal $\vec{n}_E$ of the environment map cell and $\vec{n}_L$ of the local map cell, the more similar are both cells and the higher is the distance threshold, which assigns the local cell to the background. This relation is modeled by

$$\tau = (1 - \frac{2\arccos(|\vec{n}_E \cdot \vec{n}_L|)}{\Pi})\tau^d + \tau^s$$

where τ^s and τ^d are parameters that describe the static and dynamic portion of the threshold respectively.

2) Cell classification: The cell classification is the actual step where a label is assigned to each NDT cell indicating whether it originates from a human or not. If the background subtraction in Sec. III-D.1 is enabled, the classification is applied onto the foreground map only.
The labeling is realized by a classification of the corresponding IRON descriptors with an AdaBoost classifier [21] consisting out of multiple decision trees as weak learners and trained over a set of training data.

3) Smoothing: The initial segmentation by the cell classification is not perfect as shown in Fig. 3a. Therefore, we apply an additional smoothing step in which the average of the AdaBoost classification scores of all NDT cells within the spherical neighborhood of a query cell is calculated. This new score replaces the original score of the cell afterwards. The effect of the smoothing can be seen in Fig. 3b.

4) Clustering: After having all NDT cells selected which most likely belong to a lying person, we apply DBSCAN [22], in order to cluster human-alike labeled cells close to each other into groups. We assume that all cells within a resulting cluster represent the same object. Since the cell classification in Sec. III-D.2 acts as a pre-filter, these objects are most likely humans. Without this segmentation, it could happen that the NDT map is under-segmented and, thus, results in clusters containing multiple objects in cases where they are too close to each other.

E. Cluster feature extraction

From now on, each NDT cluster is processed separately. The objective is to verify which of the found clusters really represents a fallen person. Since the clusters are usually unequal in their size and, thus, have a different number of NDT cells, appropriate features independent of the number of cells are required for this final classification. To do so, we apply an encoding of single IRON features of a cluster similar to [23] where sparse coding on 2D image descriptors was applied. All resulting code vectors of a single cluster are then combined into a single histogram representation by applying the average pooling of [23].
In particular, instead of sparse coding we do the encoding based on a soft threshold [24] on each IRON descriptor within the cluster. Although very simple, it has been shown by the authors that this is often competitive to sparse coding and thereby very fast.
Inspired by [25], we learn a set of reference IRON features with a Growing Neural Gas vector quantifier [26] in the training phase. The training as well as the encoding in the application phase is based on the cosine distance rather than the Euclidean distance in order to avoid the so called *hubness* [27] in high dimensional feature spaces.[1]

F. Mahalanobis classification

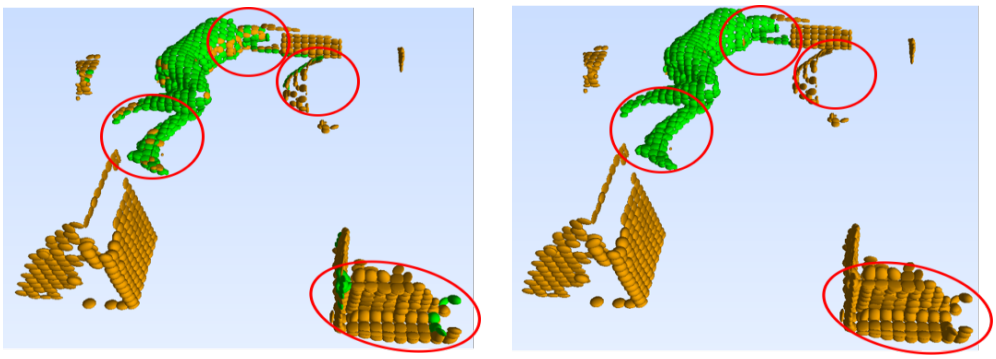
Having a histogram over all code vectors of a NDT cluster, we use a simple Mahalanobis distance and thresholding to decide, if the cluster represents a fallen person or not. The classifier is very simple but has the crucial advantage that it requires training samples from a single class only. This is an important property since the acquisition process of negative training data for a conventional classifier would be very difficult because object NDT clusters of almost all sizes and forms had to be recorded[2]. Additionally, we can generalize better in unknown environments, as we will show in Sec. IV-D.
To create the classifier, we first extract the features explained in Section III-E from all positive samples in our training set. Secondly, we use all feature vectors to compute a multivariate normal distribution. In the application phase, an unknown sample is classified via the Mahalanobis distance to the model. If it is below a certain threshold, the sample is treated as a fallen person. In order to determine an appropriate threshold, we adopt the idea from [28]: We use a test set, consisting of positive and negative samples, and calculate the probability density $p(\textit{fallen person}|d)$ which assigns a probability to be a fallen person to each distance d. With p, a suitable threshold can be indirectly defined by specifying the desired probability of misclassifications.

IV. Experiments

In order to evaluate the performance of our fallen person detector, we carried out experiments on our mobile robot presented in [3], which is a domestic health assistant. In particular, it is equipped with two laser scanners, one directed forwards and one looking backwards, an RGB-front-camera and three Asus Xtion depth cameras for sensing the area in front of the robot, the back and the ground. Since we search for persons lying on the ground, we only use the latter one

[1]The *hubness* is the effect that some feature vectors are nearest neighbor to the majority of data points.

[2]Note that this is not the case for the segmentation classifier since there just a little section of the whole NDT Map is considered which always has the same size.



(a) (b)

Fig. 3: Results of the cell classification step, where green indicates cells belonging to a human (a). Improved segmentation after the smoothing step (b). The main differences are highlighted by red ellipses.

inclined downwards for the detection task. The remaining sensors are used for obstacle avoidance and localization in our experiments. In the following, we describe our training process as well as the test data used for detecting fallen persons.[3] In the results section, we compare the proposed method with the approach based on surface normals presented in [16]. Since those features achieved the best results we do not consider the other ones studied in [16].

A. Training

There are two classifiers we need to train within our detection system: The AdaBoost classifier for the segmentation and the Mahalanobis classifier for the cluster verification. The latter only requires positive data, i.e. NDT clusters of fallen persons. Hence, we collected a set of NDT maps representing lying persons in different poses free from the ground and from any other objects. The data were captured from ten persons who lay in front of the depth sensor. Each pose was captured eight times with different orientations of the person towards the sensor respectively. We were recording continuously, while the persons moved between the different poses. That way, we were able to collect 25,150 NDT maps for the training in total.

For the segmentation classifier, we need NDT cells only. Since we use AdaBoost, additional negative samples are required which we gathered by randomly driving the robot through the regions of the environment without people and recording NDT cells not belonging to humans.

Since we did not have negative training samples of point clusters, we could not retrain the detector presented in [16] for comparison. Nevertheless, both detectors are tested on completely unknown test data recorded in our apartment-like *living lab* which should allow a fair comparison.

B. Test data

For the evaluation of the fallen person detection system, we recorded 45 test sequences while our robot was driving through an apartment-like environment autonomously. A background NDT map of the complete test environment was created beforehand. This allows the robot to locate itself within the model of the environment which is necessary in order to test the background subtraction module of our system. We recorded local NDT maps as well as the raw depth images. The latter can be converted into point clouds and were used for the reference approach in [16].

In each sequence, the robot was randomly placed within the environment and sent to a certain location, while either passing a fallen person or other objects on the ground placed to test for false alarms. In some sequences, the robot was sent to the person or a certain object directly. Per recording, one or no fallen person was present. To keep the sequences realistic, the whole drive of the robot was captured, so that other objects, furniture or other background elements, were visible, too.

The sequences have a different difficulty each. More specifically, in some cases the persons were placed in close distance to other objects, in order to make it hard to separate them from the rest of the scene. Therefore, they may touch different furniture objects, like shown in Fig. 1. Also examples with objects looking very similar to fallen persons in NDT maps were created artificially, by placing for example backpacks together with carpetbags, a vacuum cleaner and piles of blankets within the test environment. Additionally, tapes with a large dog were recorded where it was lying on the ground, sitting, standing, and walking. Furthermore, a special difficult test sequence, where the dog lies besides a fallen person, is included too. An image of it is shown in Figure 4.

For each sequence, we manually labeled a ground truth bounding box. For the evaluation, we consider the presence of a fallen person only in frames which have at least 100 NDT cells of the local NDT map within the box. In sequences without persons, the box is placed in a way that there are never cells in it. A detection is counted as positive if the centroid of the corresponding NDT cluster lies within the box. If multiple hypotheses fulfill this requirement, only

[3] Our training as well as our test data set is publicly available online at https://www.tu-ilmenau.de/neurob/data-sets-code/fallenperson/

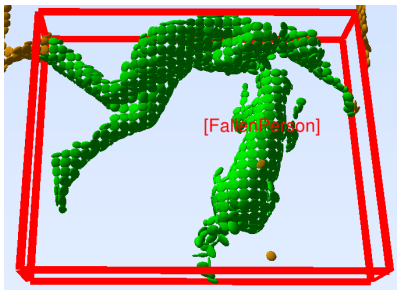


Fig. 4: Part of an NDT map of a lying person together with a dog. Both are segmented as positive within the segmentation step. Although the dog is included into the cluster wrongly, the whole cluster is classified as a fallen person. Hence, an alarm would not be misplaced in such a case. Note, that the lying dog alone is not classified as fallen person.

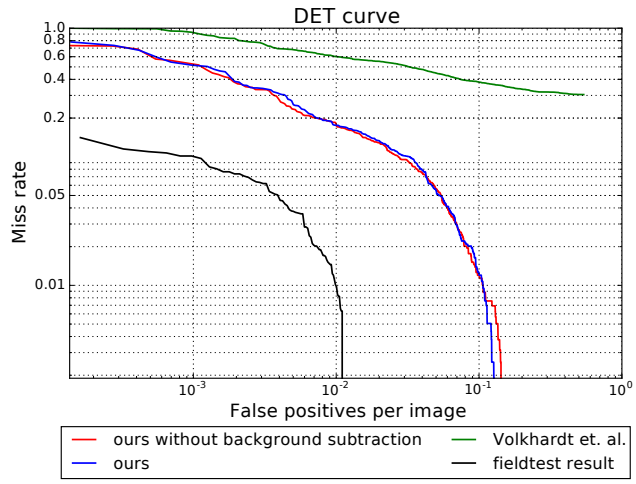


Fig. 5: Detection error tradeoff curves of our fallen person detector and of the detector presented in [16]. Both axis are in logarithmic scale. For the difference between *ours* and *ours without background subtraction* see Sec. IV-C.

the one with the highest classification score is treated as true positive, while the others are counted as false detections.
To get a fair comparison to the approach in [16], we used the same conditions for the assessment. Therefore, we additionally generated an NDT map, which is only used to decide if a ground truth example is present or not (if 100 NDT cells are within the box). This is necessary, since the method in [16] is based on point clouds.

C. Results

The results of our detector as well as from the detector in [16] are plotted as detection error tradeoff curves in Fig 5. It can be seen that our new fallen person detection system clearly outperforms the approach in [16]. We analyzed the bad performance of the latter one and found out, that the problems were caused mostly by the segmentation step. In particular, the authors used only a plane subtraction and an Euclidean clustering for segmenting persons. This cannot separate the humans from the rest of the scene good enough, especially when objects are very near by.
In contrast, our multi-stage segmentation module is able to separate fallen persons from other objects and, thus, enables a better performance for the overall detector. Our system can detect more than 80% of the persons lying on the ground while doing a single false positive detection every $100th$ frame. Most of the missed samples are scenes where a large part of a person is occluded. However, since our system can detect persons after a fall event, there should be more than one chance for a correct detection, when the robot is moving around.
Fig. 5 also shows that the background subtraction has little effect on the performance. More specifically, the performance will slightly increase if we disable the background subtraction depending on the working point in the DET curve. This can have different reasons. For our experiments, we only used a static background map. Shifted furniture releases an area where a possible fallen person would be assigned to the background. A dynamic background map like it is created when using a life-long SLAM-approach could be a solution for that problem. A second problem of the background subtraction could be cases where objects, like furniture, are touched directly. In those cases, distinctive parts of the body could be assigned to the background.
The runtime of our system is 39 ms on average on a single CPU core of an Intel i7-860 machine with eight GB RAM. This allows a detection in real-time given an update rate of ten frames per second of the depth image. Since we already use an NDT map on the robot for other tasks, the runtime of the point cloud transformation is not included in this duration. In contrast, the ground plane estimation is included, which requires a large part of the computation. If the ground plane is simply estimated by the extrinsic parameters of the sensor as described in Sec. III-A, the run time can be reduced to 16 ms. Disabling the background subtraction and estimating the ground plane by the sensor mounting results in a runtime of 18 ms, which is still real-time.

D. Field-test

To proof the generalizability and performance of our proposed detection system in a real environment, we performed a field test in an apartment of the ARTIS residential complex, situated in Erfurt (Germany). This test set consists of two scientific staff members and one female pensioner, who was still able to lie down on to the ground and get up again safely. An image of her is shown in Fig. 6. Negative test samples were generated by driving the robot through the whole apartment. It can be seen in Fig. 5 that the curve on this more realistic data is even better than that in our *living lab*. In particular, we reach about 99% correct detections by one false alarm every $100th$ frame. This can be explained with the absence of *hard to classify* examples in this very tidy apartment.

Fig. 6: Example from our field test setup. The pensioner is simulating a fall in her living room.

V. Conclusion

We have presented a novel system for detecting persons lying on the ground which can run on a mobile robotic platform in real-time. Compared to previous approaches, the major advantage is the multi-stage segmentation module based on a subtraction of the ground plane, a background subtraction, an NDT cell classification, and a clustering. Experiments have shown, that even without knowledge about the environment, i.e. without the background removal, humans can be separated almost perfectly. This allows the detection of critical fall events even in situations, where other objects are touched. The proposed approach clearly outperforms other 3D state-of-the-art fallen person detectors. By having a single false classification per 100 frames, the new system misses less than 20% of all fallen persons. In a real operational environment, the detector is even better: 99% of all fallen persons are found by producing one false alarm in 100 frames only. Since our system aims to find humans lying on the ground after a fall, it has multiple chances for the detection of an emergency case. Hence, we can use a more conservative threshold for our classifier, because we assume, that the robot can have a look onto a person from different points of view, before activating an emergency routine. This is part of an ongoing work dealing with integrating the individual detection results in a more consistent overall decision of a fall situation.

References

[1] A. Icks, "Risikofaktor Stürze im Alter," *BKK Gesundheitsreport 2013*, November 2013, 77–80.

[2] S. Bundesamt, "Bevölkerung Deutschlands bis 2060 – 13. Koordinierte Bevölkerungsvorausberechnung," *Statistisches Bundesamt: Wiesbaden*, 2015.

[3] H.-M. Gross, St. Mueller, Ch. Schroeter, M. Volkhardt, A. Scheidig, K. Debes, K. Richter, and N. Doering, "Robot companion for domestic health assistance: Implementation, test and case study under everyday conditions in private apartments," in *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2015, 5992–5999.

[4] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3d-ndt," in *Journal of Field Robotics*, vol. 24, no. 10, 2007, 803–827.

[5] X. Yu, "Approaches and principles of fall detection for elderly and patient," in *Int. Conf. on e-Health Networking, Applications and Services (HealthCom)*, 2008, 42–47.

[6] N. Noury, P. Rumeau, A. Bourke, G. ÓLaighin, and J. Lundy, "A proposal for the classification and evaluation of fall detectors," in *Journal of the AGBM (IRBM)*, vol. 29, no. 6, 2008, 340–349.

[7] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches," in *Neurocomputing (NEUCOM)*, vol. 100, 2013, 144–152.

[8] R. Igual, C. Medrano, and I. Plaza, "Challenges, issues and trends in fall detection systems," in *Biomed. Eng. Online*, vol. 12, no. 66, 2013, 1–66.

[9] S. Georgakopoulos, S. Tasoulis, I. Maglogiannis, and V. Plagianakos, "On-line fall detection via mobile accelerometer data," in *Int. Conf. on Artificial Intelligence Applications and Innovations (AIAI)*, 2015, 103–112.

[10] I. N. Figueiredo, C. Leal, L. Pinto, J. Bolito, and A. Lemos, "Exploring smartphone sensors for fall detection," in *mUX: The Journal of Mobile User Experience*, vol. 5, no. 1, Springer, 2016, 1–17.

[11] A. Yajai, A. Rodtook, K. Chinnasarn, and S. Rasmequan, "Fall detection using directional bounding box," in *Int. Joint Conf. on Computer Science and Software Engineering (JCSSE)*, 2015, 52–57.

[12] G. I. Parisi and S. Wermter, "A neurocognitive robot assistant for robust event detection," in *Trends in Ambient Intelligent Systems*. Springer, 2016, 1–27.

[13] S. Wang, S. Zabir, and B. Leibe, "Lying pose recognition for elderly fall detection," in *Robotics: Science and Systems VII*, no. 1, MIT Press, 2012, 345–353.

[14] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008, 1–8.

[15] D.-X. Xia, S.-Z. Su, S.-Z. Li, and P.-M. Jodoin, "Lying-pose detection with training dataset expansion," in *Int. Conf. on Image Processing (ICIP)*, 2014, 3377–3381.

[16] M. Volkhardt, F. Schneemann, and H.-M. Gross, "Fallen person detection for mobile robots using 3d depth data," in *Int. Conf. on Systems, Man, and Cybernetics (SMC)*, 2013, 3573–3578.

[17] T. Schmiedel, E. Einhorn, and H.-M. Gross, "IRON: A fast interest point descriptor for robust NDT-map matching and its application to robot localization," in *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2015, 3144–3151.

[18] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM (CACM)*, vol. 24, no. 6, 1981, 381–395.

[19] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation," in *Intelligent Vehicle Symposium*, vol. 2, 2002, 646–651.

[20] E. Einhorn and H.-M. Gross, "Generic NDT mapping in dynamic environments and its application for lifelong slam," *Journal of Robotics and Autonomous Systems (RAS)*, vol. 69, 2015, 28–39.

[21] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *European Conf. on computational learning theory (EuroCOLT)*, 1995, 23–37.

[22] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Int. Conf. on knowledge discovery and data mining (KDD)*, 1996, 226–231.

[23] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009, 1794–1801.

[24] A. Coates and A. Y. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2011, 921–928.

[25] K. Labusch, E. Barth, and T. Martinetz, "Learning data representations with sparse coding neural gas." in *European Symposium on Artificial Neural Networks (ESANN)*, 2008, 233–238.

[26] B. Fritzke *et al.*, "A growing neural gas network learns topologies," *Advances in neural information processing systems (NIPS)*, vol. 7, 1995, 625–632.

[27] M. Radovanović, A. Nanopoulos, and M. Ivanović, "Hubs in space: Popular nearest neighbors in high-dimensional data," *Journal of Machine Learning Research (JMLR)*, vol. 11, 2010, 2487–2531.

[28] M. Eisenbach, A. Kolarow, A. Vorndran, J. Niebling, and H.-M. Gross, "Evaluation of multi feature fusion at score-level for appearance-based person re-identification," in *Int. Joint Conf. on Neural Networks (IJCNN)*, 2015, 1–8.