Recursive Weighted 2-Means Split Algorithm For Under-segmentation Reduction

Kornelija Magylaitė [1], Lukas Arlauskas [2] and Karolis Ryselis [3]

[1] *Software Engineering Department, Kaunas University of Technology*
[2] *Information Systems Department, Kaunas University of Technology*
[3] *Software Engineering Department, Kaunas University of Technology*

Abstract

Human body segmentation is utilized in various applications as an intermediate step. This problem is best solved using supervised machine learning solutions, however, they require annotated data for training. Unfortunately, annotating data for segmentation is an extremely labor-intensive task. It could be improved by using semi-automatic segmentation algorithms, however, their accuracy tends to be low for complex scenes. The errors made by such algorithms can be manually corrected by a human. This process could be more efficient when automatic correction tools are added to the data processing pipeline. This research aims to improve the final semi-automatic segmentation accuracy by improving an existing random forest classifier for correcting point cloud segmentation based on metrics of recursive 2-Means split. We replace the K-Means clustering with a weighted K-Means clustering and optimize the weights. Experiments have revealed that the segmentation accuracy of 62.4% is improved to 66.1% with a weight ratio of 0.4. Since higher accuracy leads to less manual labor, this is a sought-after improvement that reduces the time to prepare datasets for human body segmentation.

Keywords

weighted K-Means, human body segmentation, random forest

1. Introduction

Depth data processing has been an active research subject in the recent times, because of its' plentiful application ways. The spatial data is being used in various spheres from depth cameras to lidars. However, machine learning models used for data processing, require a lot of depth data. The depth data can be extracted from depth sensing devices, however manual segmentation and annotation is rather repetitive process and includes a huge amount of manual labour. For example, "Kinect" sensor produces 30 depth frames per second. Therefore, it is not possible to effectively segment the data using manual methods. Trying to alleviate the problem, there has been research in trying to make the process as automatized as possible. The research conducted by one of the authors of this article [1] includes implementation and experimentation with Point cloud library [2]. The experiment was previously made using specifically K-Means algorithm and a random forest classifier. While the classifier achieved a quite high success rate, the accuracy of bounding-box-based segmentation was significantly lower. Implementation of the 1:1 cuts methodology resulted in a subsequent improvement. Nevertheless, the algorithm exhibited poor efficacy when presented with complex and combined datasets, as it failed to accurately trim the edges of the image. As such, the goal of this research is to improve the existing solution's accuracy in recognizing complex poses while maintaining non-worsening accuracy in detecting simple poses. The paper is structured in a following format. Section II discusses principles of weighted K-Means algorithm. Section III describes the problem and research methodology as well as

28th Conference on Information Society and University Studies (IVUS'2023), May 12, 2023, Kaunas, Lithuania
EMAIL: kristina.magylaite@ktu.lt; lukas.arlauskas@ktu.lt; karolis.ryselis@ktu.lt

algorithm modifications in detail. Section IV provides the results of accuracy and discusses the evaluation of the algorithm. Finally, Section V concludes the article.

2. Related Work

2.1. Random Forest Classifier

Random forest is a supervised machine learning technique [3]. It is a collection of decision trees that vote on the final decision. It is meant for classification problems. Random forests can be applied to segmentation tasks. However, in this case, features for training are required. These features are extracted in different ways and can be hand-crafted or computed by other machine-learning systems. Hand-crafted features are wildly various in the state of the art. They are mostly domain-specific features like elevation data [4], coordinates in space [5] or pixel intensity [6]. However, they are always different and depend on the task at hand.

2.2. Weighted K-Means Algorithm

K-Means algorithm is an unsupervised learning technique for clustering [7]. It has a hyperparameter k which represents the number of clusters in the output. However, the original algorithm uses equal weights for all centroids. There are many variations of the algorithm that try to introduce different weights for the clusters. They are successfully applied for image segmentation for RGB image segmentation [8] or image clustering [9]. This algorithm is useful in the cases where different classes of objects should have different sensitivities. It translates to assigning weights to the centroids. On the other hand, it is only possible when the area of application is known in advance as the weights are derived from the area of application.

There are several challenges when applying the weighted K-Means algorithm. First, the centroids must have different weights based on the area of application. It is solved very differently in the state of the art – different features may be pre-selected [10], variable relative relevance may be estimated [11], the weights may even be acquired using machine learning solutions [12]. Unfortunately, there is no single best way to solve this problem.

2.3. Random Forest Classifier for Correcting Point Cloud Segmentation Based on Metrics of Recursive 2-Means Split

Semi-automatic segmentation algorithms tend to make under-segmentation errors. One of the ways to solve this problem is cutting parts of the resulting point cloud. This can be achieved by using K-Means algorithm adaptation suggested in the state of the art [1]. The research proposes the following workflow:

1. The segmented output is split into two clusters using K-Means algorithm with hyperparameter k = 2;
2. 8 metrics are computed for the resulting split:
    a) All mean distances between the combinations of both centroids and three clusters (initial cluster and the new sub-clusters) (six distances);
    b) Sizes of the new clusters (two sizes);
3. A random-forest-based classifier predicts the quality of the split based on the computed metrics and outputs the probability that rejecting one of the clusters improves the segmentation quality;
4. If the quality is improved, the second cluster is rejected and the process is recursively repeated from step 1.

The first step involves a modification of K-Means algorithm. The initial points are selected by a human. After the first points are being set, it is known that they are already correct. These provided

points are fixed centroids that never move. The point that is the furthest away from the first centroid is chosen as a second centroid. Now that the second centroid is being positioned, it is being updated until it converges, but the improvement is limited up to 10 times due to the long runtime prevention.

The metrics used in the second step were chosen based on the hypothesis that their combined values can potentially indicate different cluster types. For instance, if the original segmentation identifies a single object, the distances between both point clouds and centroids will be significantly lower than if the two clusters represent two distinct objects. Similarly, if the entire background is captured, the false cluster will be much larger than the true cluster. If two very distinct objects are in the cluster, the average distances between subclusters and their centroids will be considerably smaller than the average distance between the sub-clusters and the other centroid. The advantage of these metrics is that they can be computed relatively quickly.

The classifier of the third step is trained on the computed metrics. The data consists of the metrics and a flag that indicates whether the split improves the accuracy or not. The original research utilizes a self-acquired and self-annotated dataset to generate the training data. This dataset was also available during the experiments presented in this article.

3. Methodology

3.1. The Problem

While the presented K-Means algorithm achieves 95% accuracy for the classifier, the primary objective is to enhance the accuracy of the segmentation process. Currently, the bounding-box-based segmentation approach achieves a 33% accuracy rate. By implementing the 1:1 cuts methodology, the accuracy rate has improved to 55%. However, further enhancements are needed beyond this level of improvement. It is currently incapable of trimming smaller edges of the image. In the worst-case scenarios, the entire scene is enclosed in a single segment, while the human element only comprises roughly 10% of the pixels. If cuts are made using a 1:1 ratio, the first cut will leave behind 50% of the scene, the second cut - 25%, and the third cut would only succeed in an ideal scenario where the entire human element fits into a single cluster. In instances where the human is positioned closer to the center, only the edges need to be trimmed, and there is no single split with a 1:1 sensitivity that can accomplish this. As shown in Fig. 1, two examples represent outputs of primary algorithm. Red color depicts under-segmentation, green – over-segmentation errors, yellow – correct segmentation output.

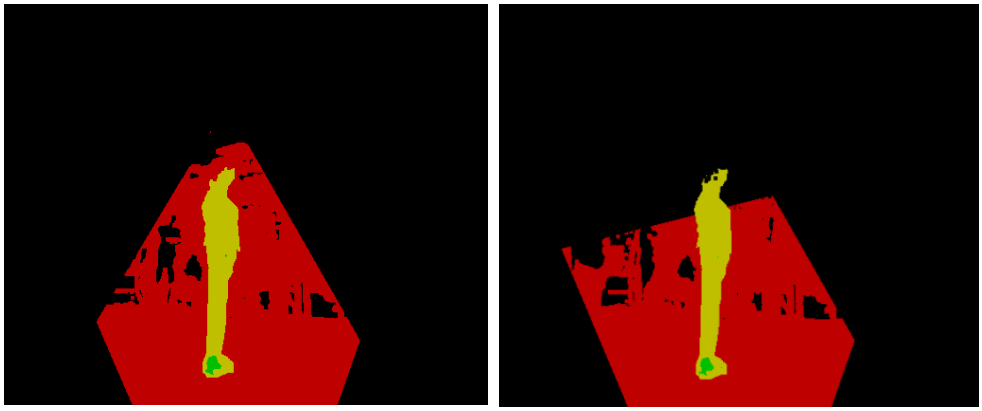

Figure 1: Example of the edges not being trimmed correctly

3.2. Application of Weighted K-Means

The purpose of assigning different weights to the centroids is to improve the partitioning of data points. Decreasing weight of the centroid makes the other centroid more sensitive and, consequently,

more data points are assigned to the centroid with lower weight, which makes the algorithm more efficient and adaptable to different datasets. For instance, in cases where the human is positioned closer to the center, only the edges require trimming, and using different sensitivity ratios may produce better results. However, a lower sensitivity ratio implies that more splits are necessary, which demands higher classification accuracy as there will be more classifications needed. The main challenge is to determine the most optimal sensitivity ratio that achieves the best accuracy.

3.3. Finding Proper Weight

To determine the required ratio of sensitivity, an optimal proportion of weight had to be identified. The objective was to find a proper ratio of sensitivities for centroids in order to achieve higher accuracy. The ratio of sensitivities can be described as variable relative relevance, as the variables are the centroids selected by a human and the potential background centroid. The examination of variable relevance is crucial, given the previous algorithm's inadequate performance with specific datasets.

The ratio of weights was chosen from the range of (0; 1) with a step of 0.1. To decrease the sensitivity of a centroid, a lower weight was assigned to it. This indicates that the multiplication between the calculated distance and the fixed weight resulted in a lower distance for the centroid. As a result, it is more probable that the point will be assigned to the centroid with lower weight. For instance, if the ratio of the weights is 1:2, it implies that a point must be twice as close to the second centroid to be assigned to it. Otherwise, it will be assigned to the first one.

However, the presented example is purely theoretical. The weights ratio may vary, as well as the selection of centroids with varying levels of sensitivity. To identify the optimal ratio that achieves the desired accuracy, an experiment was performed. The process involved testing the algorithm using several ratios within a predetermined range.

3.4. Workflow of Testing

The presented methodology in Fig. 2 illustrates the procedural steps involved in the testing process for a modified algorithm. The process comprises four primary actions aimed at effectively testing the algorithm's performance.

Initially, the desired weights were set to determine the centroid sensitivities. This was achieved by considering previous weights and their corresponding outcomes. Additionally, the goal was to test all possible combinations to identify the most optimal solution.

Subsequently, centroid sensitivities were set, and data was generated for the classifier. The data used for the generation consists of self-acquired datasets containing depth images of people. The datasets have been acquired using "Kinect" sensor. They have been labelled semi-automatically with the help of solution without random forest accuracy improvements [1]. After the generation a dataset was created that would be utilized to train and evaluate the performance of a random forest classifier algorithm. The data generation was implemented using the Java programming language (OpenJDK 14).

The generated data was then used for training, as mentioned previously. The random forest classifier was selected as the algorithm of choice since it typically produces superior outcomes compared to a single tree. The implementation of this random forest classifier was carried out using the Python programming language, specifically utilizing the sklearn library.

The segmentation accuracy finally assessed by obtaining mean accuracies for simple, complex and combined datasets.

Figure 2: Workflow of testing

4. Experimental Evaluation of the Algorithm

The algorithm was evaluated experimentally using the techniques described in the related work section. The results constitute mean segmentation accuracy. Mean segmentation accuracy is measured by calculating a cross-set intersection coefficient (see (1)):

$$a = \frac{n(A \cap G)}{n(A)} \times \frac{n(A \cap G)}{n(G)} \quad (1)$$

where A is the set of points selected during the segmentation, G is the set of ground truth points.

For the hypothesis to be confirmed it was important that when using weighted centroids, the results with simple dataset would not worsen and would improve over baseline (id est the variant with second centroid weight of 1) with complex and combined datasets. The experiment involved assigning lower weight to both centroids at different times. The results indicated that superior results were obtained when assigning lower weight to the second centroid. When comparing the results obtained using different second centroid weights (see Table I), the best results were a mean accuracy of 82.94% (0.5% improvement over baseline) while using simple dataset, mean accuracy of 59.62% (7.8% improvement over baseline) while using complex dataset and combined dataset mean accuracy of 66.05% (5.75% improvement over baseline). The best results for simple dataset were achieved with second centroid weight of 0.5, while the best accuracy for the complex and combined mean accuracy results were obtained with the second centroid weight of 0.4. These solutions use desensitized first centroid which means more points get attributed to the second centroid. Interestingly, more than half of the results obtained with the simple dataset were rather close to the best solution (weights of 0.1 through 0.4), something that cannot be said about the complex and combined datasets accuracy results.

These findings are further reinforced when comparing the results of random forest classification reports with different second centroid weights (see Table II). The best precision, recall and F1-score for correct cuts are 0.96, 0.97 and 0.96 respectively while the best results using the same metrics for incorrect cuts are 0.98, 0.97 and 0.98 respectively. The best accuracy achieved was 0.97. The consistently triumphant variant was the one with second centroid weight of 0.4, although the variant with 0.5 is also close call.

Table 1

Algorithm accuracy comparison using different values for second centroid. Note: bold denotes the maximum value of the column.

Second centroid weight	Simple dataset accuracy	Complex dataset accuracy	Combined datasets mean accuracy
1 (baseline)	82.5%	55.3%	62.46%
0.1	82.2%	47.62%	57.15%
0.2	82.12%	44.85%	55.15%
0.3	82.12%	44.75%	55.07%
0.4	82.9%	**59.62%**	**66.05%**
0.5	**82.94%**	50.95%	59.79%
0.6	81.73%	42.72%	53.49%
0.7	80.96%	44.89%	54.85%
0.8	80.88%	40.94%	51.97%
0.9	80.32%	48.4%	57.21%

Table 2

Random forest classification reports with different second centroid weights comparison. Note: bold denotes the maximum value in that group of values.

Second centroid weight	Precision	Recall	F1-score

Correct cuts – weight of 1	0.94	0.90	0.92
Correct cuts – weight of 0.4	**0.96**	**0.97**	**0.96**
Correct cuts – weight of 0.5	**0.96**	**0.97**	**0.96**
Incorrect cuts – weight of 1	0.94	0.90	0.96
Incorrect cuts – weight of 0.4	**0.98**	**0.97**	**0.98**
Incorrect cuts – weight of 0.5	0.97	0.96	0.96
Accuracy – weight of 1	0.95		
Accuracy – weight of 0.4	**0.97**		
Accuracy – weight of 0.5	0.96		

When taking a look at box plots, it becomes apparent that when using 0.4 as second centroid weight, median accuracy of the algorithm nears 100% in specific cases and further solidifies the findings that algorithm accuracy didn't worsen by using weighted centroids in the case of simple dataset (see Fig. 3 and Fig. 4). The accuracy box plot while using complex dataset reveals such comparative differences: the accuracy improved as the accuracy data contained in the second and third quartiles got pushed up and the upper half of data is contained within less space. The front poses accuracy got improved as well as adding weights got rid of all outliers at the cost of more sparsely spaced out predictions. Accuracy improvements can be observed in the side parts accuracy (see Fig. 5 and Fig. 6).

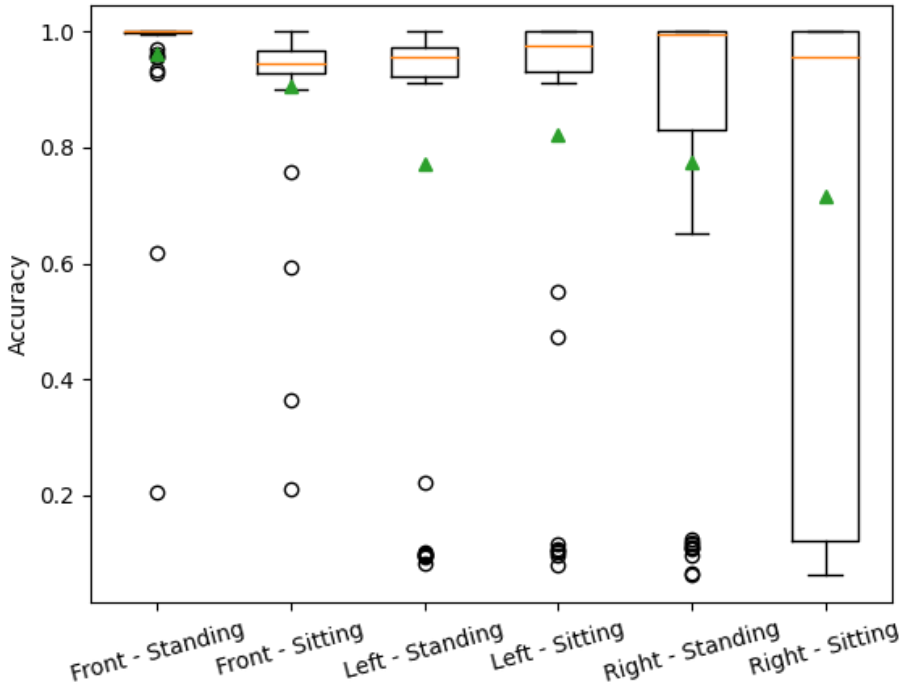

Figure 3: Accuracy box plot using simple dataset (without weights)

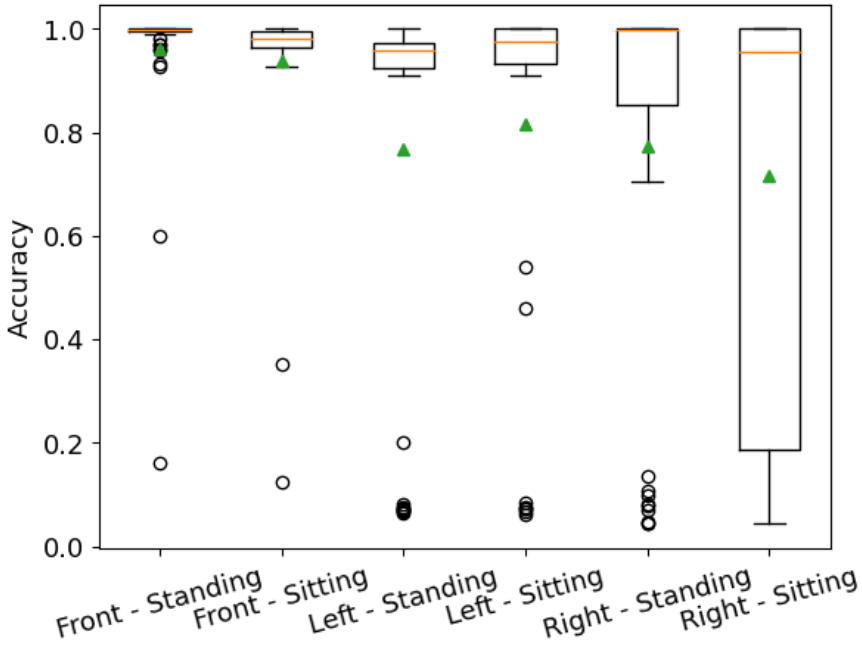

Figure 4: Accuracy box plot using simple dataset and 0.4 second centroid weight

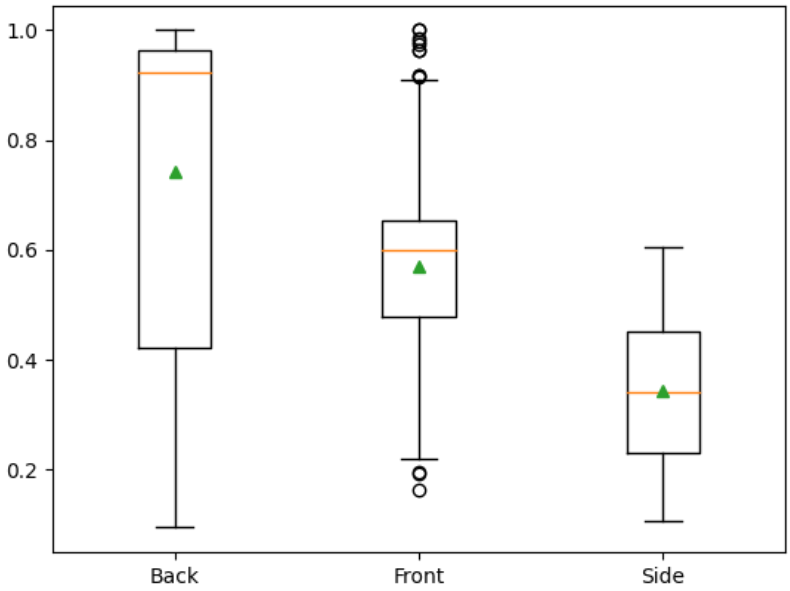

Figure 5: Accuracy box plot using complex dataset (without weights)

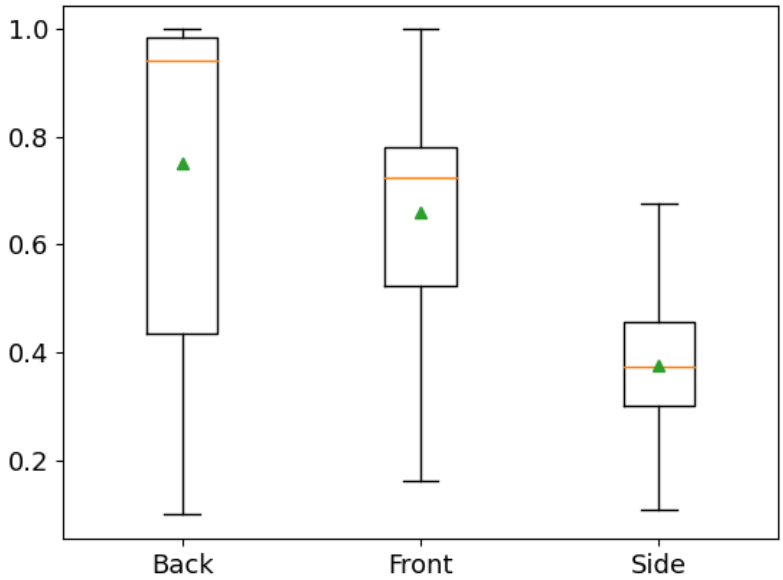

Figure 6: Accuracy box plot using complex dataset and 0.4 second centroid weight

Figure 7 demonstrates several mistakes made by the proposed algorithm. Red color depicts under-segmentation, green – over-segmentation errors, yellow – correct segmentation output. In all cases the under-segmented area is positioned around the human. The last three examples show that the cluster weights should be very different to cut such small pieces on either side of the human body. Unfortunately, this is limited by the classifier accuracy – very low weight ratio means many splits and many predictions. Since the classifier accuracy is currently 97%, making 20 splits would mean that the probability that all of them are correct is only about 54%. Despite that, the manual work required to remove the red areas of the images is lower than presented in the original research [1].

The second image shows an error that occurs due to the centroid positioning. The algorithm decided that cutting the legs of the human adds accuracy compared to leaving too many pixels in the output, which is true – this leads to different but smaller mistake.

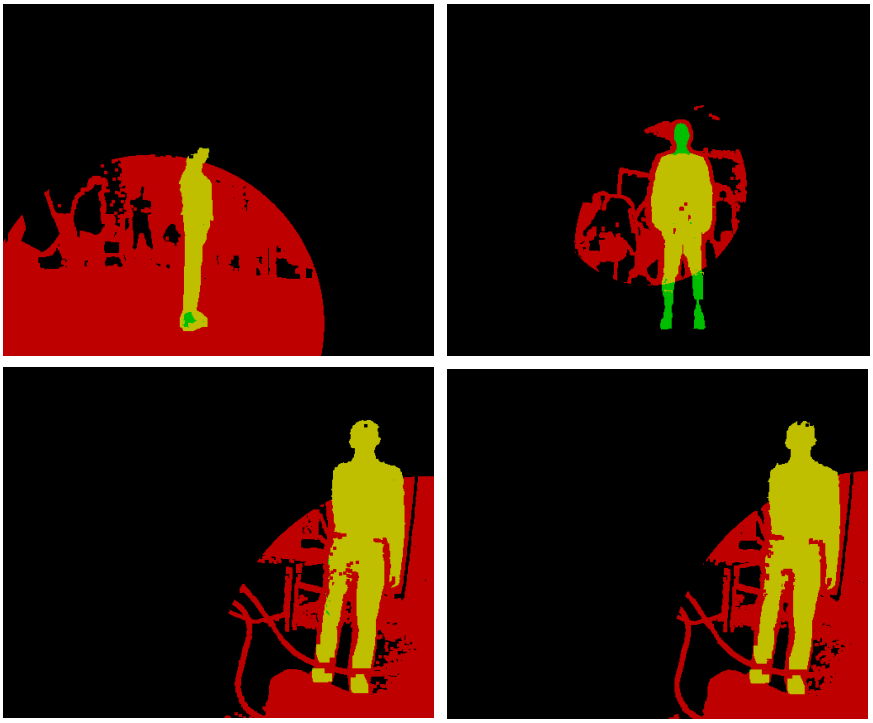

Figure 7: Examples of erroneous segmentation output

5. Conclusions

In the experiments performed in the paper, several weighted K-means variants consistently outperformed the baseline unweighted one. Using 0.5 as second centroid weight, simple dataset accuracy of 82.94% was achieved. The complex and combined datasets benefited the most from 0.4 as second centroid weight. The further research on classification report then revealed that the precision, recall and f1-score were tied between the 0.4 and 0.5 variants on the correct cuts, however as far as the incorrect cuts and overall accuracy is concerned, the variant with 0.4 weight prevailed in all the metrics.

Our research demonstrated that for human body segmentation, it is better to use weighted K-means algorithm as it seemingly gets rid of outlier data as revealed by the box plots and the weighted variant can be further applied without sacrificing any important qualities of the baseline algorithm. The further research that could be done is further optimization of the algorithm for the complex and combined datasets with the aim of improving the accuracy of segmentation.

6. References

[1] Ryselis, "Random Forest classifier for correcting point cloud segmentation based on metrics of recursive 2-means splits," in Information and Software Technologies: 28th International Conference, ICIST 2022, Kaunas, Lithuania, October 13–15, 2022, Proceedings. Springer, 2022, pp. 90–101.

[2] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," Ph.D. dissertation, Computer Science department, Technische Universitaet Muenchen, Germany, 10 2009.

[3] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp.5–32, 2001.

[4] H. Ni, X. Lin, and J. Zhang, "Classification of als point cloud with improved point cloud segmentation and random forests," Remote Sensing, vol. 9, no. 3, p. 288, 2017.

[5] S. Pereira, A. Pinto, J. Oliveira, A. M. Mendrik, J. H. Correia, and C. A.Silva, "Automatic brain tissue segmentation in MR images using random forests and conditional random fields, "Journal of neuroscience methods, vol. 270, pp. 111–123, 2016.

[6] Y. Wu and S. Misra, "Intelligent image segmentation for organic-richshales using random forest, wavelet transform, and hessian matrix," IEEE Geoscience and Remote Sensing Letters, vol. 17, no. 7, pp. 1144–1147, 2019.

[7] J. MacQueen, "Classification and analysis of multivariate observations," in 5th Berkeley Symp. Math. Statist. Probability. University of California Los Angeles LA USA, 1967, pp. 281–297.

[8] A. Shmmala and W. Ashour, "Color based image segmentation using different versions of k-means in two spaces," Global Advanced Research Journal of Engineering, Technology and Innovation, vol. 1, no. 9, pp.030–041, 2013.

[9] Yu, S. A. Velastin, and F. Yin, "Automatic grading of apples based on multi-features and weighted k-means clustering algorithm," Information Processing in Agriculture, vol. 7, no. 4, pp. 556–565, 2020.

[10] M. W. Ayech and D. Ziou, "Terahertz image segmentation using k-means clustering based on weighted feature learning and random pixel sampling," Neurocomputing, vol. 175, pp. 243–264, 2016.

[11] A. K. Gupta, A. Seal, P. Khanna, O. Krejcar, and A. Yazidi, "Aw k s: adaptive, weighted k-means-based superpixels for improved saliency detection," Pattern Analysis and Applications, vol. 24, pp. 625–639,2021.

[12] N. Ohana-Levi, I. Bahat, A. Peeters, A. Shtein, Y. Netzer, Y. Cohen,and A. Ben-Gal, "A weighted multivariate spatial clustering model to determine irrigation management zones," Computers and Electronics in Agriculture, vol. 162, pp. 719–731, 2019.