Ben-Gurion University of the Negev

Faculty of Engineering Science

School of Electrical and Computer Engineering

Dept. of Electrical and Computer Engineering

Fourth Year Engineering Project

Final Report

Tracking objects in 3D with the help of integral imaging and deep learning

Project number: p-2024-031

Students: Yair Ross – 207287889

Moshe Shmilovich – 208583948

Supervisor: Professor Yitzhak Itzhaki

Submitting date: 18-08-2024

תקציר:

בפרויקט זה נשתמש במטריצת מצלמות (3X7 מצלמות) שנמצא במעבדה ונצלם קטעי וידאו קצרים של מספר שניות מצומצם שבהם רואים עצם יחיד או מספר עצמים שנעים במרחב.

המצלמות הינן מצלמות פשוטות באיכות בינונית שכל אחת מצלמת את הוידאו מזווית קצת שונה.

מטרתנו הינה לבנות מערכת אשר מקבלת קטע וידאו קצר של מערך המצלמות ותדע לזהות את כל העצמים באמצעות סגמנטציה, לזהות את מרחקם, לסמן את כולם ולעקוב אחר עצמים אלו. כלומר ברגע שהמערכת זיהתה עצם בסרטון היא תצטרך במהלך הפריימים הבאים לזהות את אותו עצם ולסמן שזהו אותו עצם. זאת נעשה על ידי אלגוריתם עקיבה שנממש במהלך הפרויקט.

אנו נתמקד בהרחבת המערכת הקיימת כבר של זיהוי מרחקי אובייקטים כדי להיות מסוגלים לעקוב אחר אותם אובייקטים באמצעות המידע של המרחקים והמאפיינים השונים של אותם אובייקטים באמצעות אלגוריתם שאנו מעצבים.

Abstract:

In this project we will use a camera matrix (3X7 cameras) located in the laboratory and we will take short video clips of a limited number of seconds in which you see a single object or several objects moving in space.

The cameras are simple, medium quality cameras that each shoot the video from a slightly different angle.

Our goal is to build a system that receives a short video clip of the camera array and will know how to identify all the objects through segmentation, identify their distance using an approach based on computational integral imaging, mark them all and track these objects. That is, once the system has identified an object in the video, it will have to identify the same object during the following frames and mark it as the same object. This is done by a tracking algorithm that will be implemented during the project.

We will focus on expanding the already existing system of identifying objects distances to be able to track those objects using the information of the distances and different properties of those objects using an algorithm which we design.

Table of Contents:

1. Theoretical background

1.1. Our goal in the project:

We will take the existing system and want to add object tracking capabilities. This means that the system will know for each frame where our object moves and re-identify it and not make the mistake of thinking that it is another object.

The method in which we will try to realize this is by changing the approximate position of the object and finding the object in the new field and comparing it to the object in the previous frame by finding similar characteristics.

Our goal is that the system will manage to track all objects in a video for all of its frames and our success will be measured by how many frames it manages to do so in a video and how many videos it will work on

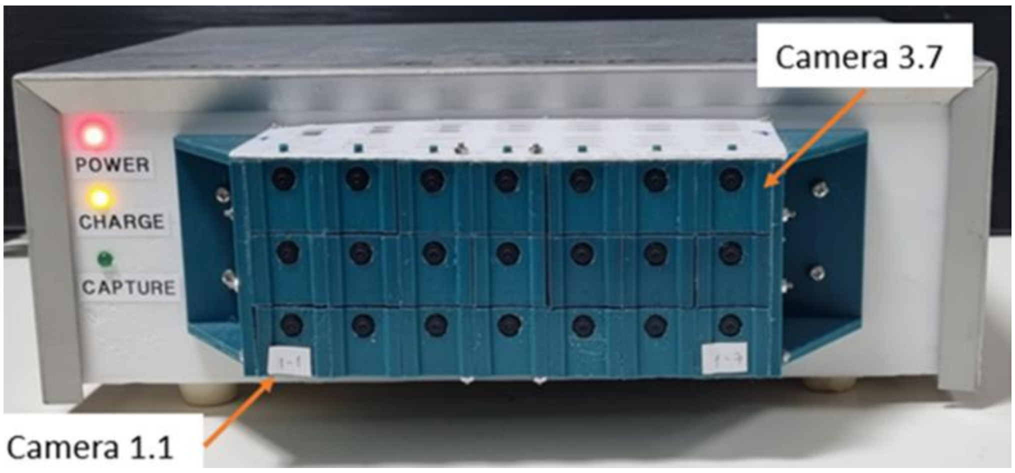


Figure 1. Image of the camera array

1.2. <u>Object detection in the picture:</u>

In an article that performs object detection in 3D using established 2D segmentation Computational Integral Imaging (CII) on video clips [1] is reviewed in detail a method for detecting objects in space and finding their distance from the camera lens. We will briefly review the above detailed method.

The method on which the solution is based is called Computational Integral Imaging. In this method, 3D objects are reconstructed either optically or computationally.

The reconstructed depth plane of the imaging system in depth z_0 For an array of elementary images calculated by:

$$f^{RP}(x,y,z_0) = \frac{1}{KL}\sum_{k=0}^{K-1}\sum_{l=0}^{L-1} g_{k,l}\left(x + \left(\frac{1}{M_{z_0}}\right)S_x k, y + \left(\frac{1}{M_{z_0}}\right)S_y l\right) \tag{1}$$

Where:

-$g_{k,l}$ is an elemental image

-k and l indices

-K × L are the number of EIs in the array

-M is the magnification factor that the ratio between the distance from the camera to the reconstructed plane and the camera's focal distance.

-s_x are the distances between the cameras in the x direction.

-s_y are the distances between the cameras in the y direction.

-$f^{RP}(x,y,z_0)$ is the 2D reconstructed plane at a distance z_0 from the camera.

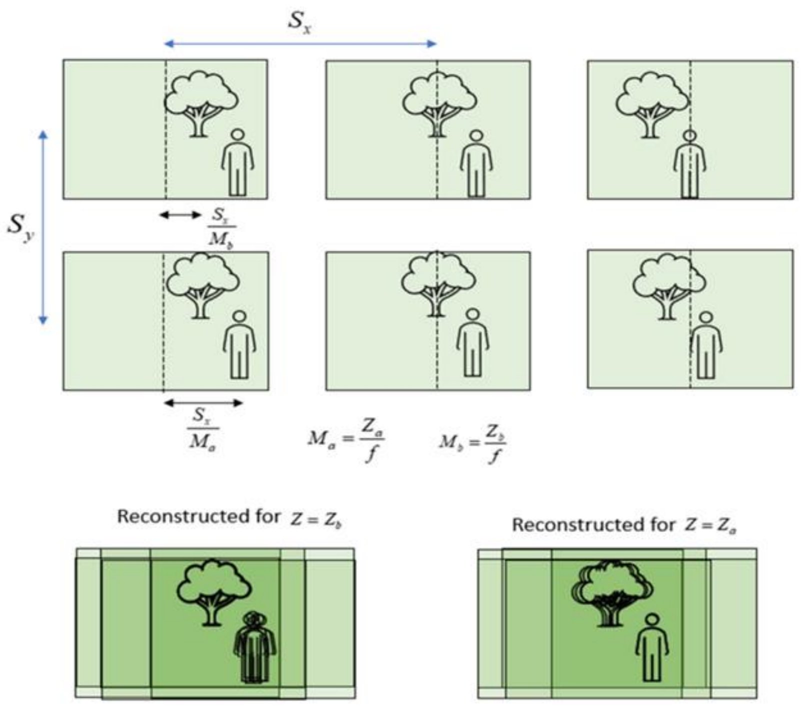


Figure 2. Illustration of the reconstruction of the depth plane at two different distances. Above we see six elementary pictures which were taken at slightly different angles, and below you can see the reconstruction of the plane at the depth of the tree and at the depth of the person, and in each of them the bone that is at the appropriate depth looks sharp compared to the other bone that comes out blurry.

After constructing the reconstructed plane, the magnitude gradient is calculated of it. In a plane at a depth corresponding to a certain bone, the bone came out in the reconstruction in a sharper way, therefore the gradient in this plane will be higher. The average gradient for a reconstructed plane at depth z is calculated by:

$$AGMR(z) = \frac{1}{N_x N_y} \sum_y \sum_x \left| \nabla \left(f^{RP}(x,y,z) \right) \right| \tag{2}$$

Where:

$-AGMR(z)$ is the Average Gradient Magnitude of a Reconstructed plane at depth z.

$-N_x$ are the numbers of pixels along the x direction.

-N_y are the numbers of pixels along the y direction.

-∇ is the gradient magnitude operator.

The gradient is calculated for planes at different distances from the lens. We can get a graph so that the axis of X is the distance and the Y axis is the gradient and we will get a function whose maximum point is the distance where there is an object that is sharply reproduced and this is the approximate distance of the object. The sharp objects are isolated from the blurred parts of the image by finding a threshold value on the magnitude gradient of the reconstructed plane.

The disadvantages of this method are that it has difficulty identifying small objects and is also unable to differentiate between objects at a similar distance and it is difficult to define the threshold value.

<u>1.3. The improved method of object detection in 3D [1]:</u>

The method implemented in practice uses the array of cameras (of 3 x 7) to take photos and videos, with each camera at a slightly different angle. The algorithm uses each frame of the video using the method MASK-R-CNN [2] for detecting and identifying objects in the image and delineating them in a bounding box. After identifying the objects and delimiting them, we will activate a depth plane reconstruction on each object in the area where the object is bounded at different depths (similar to the previous method [3] but this time in a focused area of the object). An AGMR calculation is then run on each plane to find the depth position of the object from the lens.

The improved method of 3D object detection diagram (2):

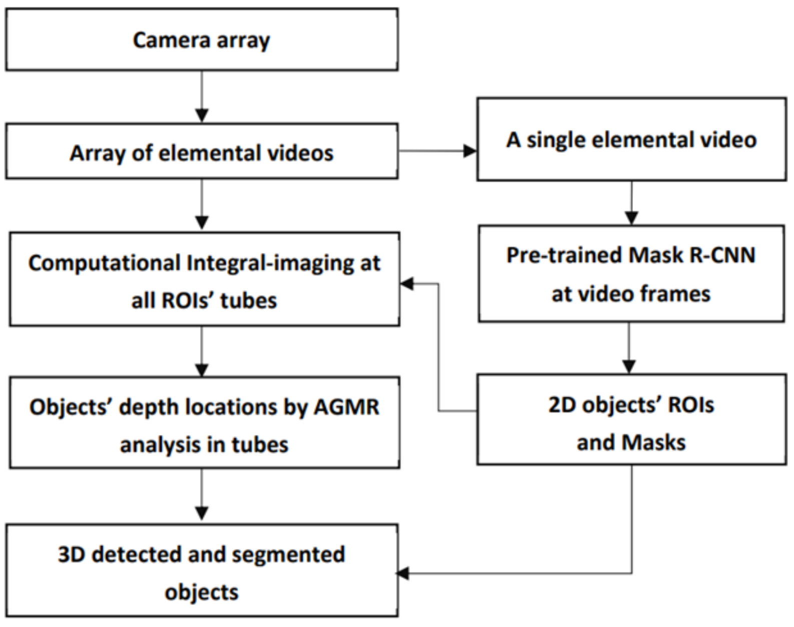

Camera array
Array of elemental videos
A single elemental video
Computational Integral-imaging at all ROIs' tubes
Pre-trained Mask R-CNN at video frames
Objects' depth locations by AGMR analysis in tubes
2D objects' ROIs and Masks
3D detected and segmented objects

Figure 3. Diagram of the improved method for detecting objects in 3D

The improvement compared to the previous method is the solution of the problems that were:

- Difficulty in identifying depth of small objects
- Difficulty identifying two objects at the same depth

These problems are solved by creating a tube of reconstructed depth planes which are localized only to the region of the bounding box of a certain object. A calculation of the magnitude of the gradient will be made on these planes to find a maximum value that will determine the true distance of the object. With this method, the AGMR calculation will not be affected by other objects in the image and will be focused only on the relevant object, therefore it solves the problem of small objects. In addition, for objects at the same depth, a different calculation will be performed for each of them for pipes of different local depth planes. A modification of the AGMR formula is necessary to calculate the magnitude of the gradient only in the appropriate area and the formula is:

$$AGMR_k^{O_i}(z) = \frac{1}{N_x N_y} \sum_{y \in ROI_i} \sum_{x \in ROI_i} \left| \nabla \left(f_k^{RP}(x,y,z) \right) \right| \tag{3}$$

To find the depth we find the index with the maximum AGMR and the Z value at this point is the depth chosen for the object:

$$z_k^{O_i} = \text{argmax}\left(AGMR_k^{O_i}(z) \right) \tag{4}$$

We can see an example of this process:

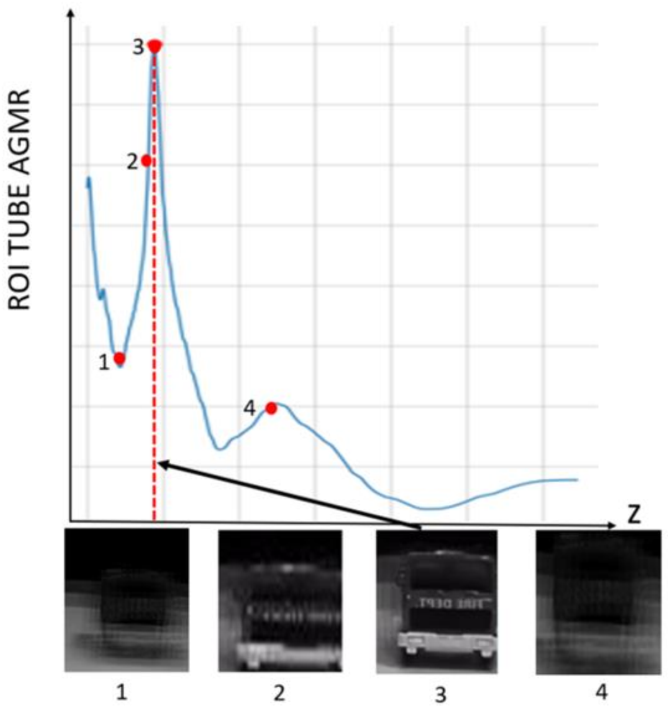


Figure 4. Shows that for four different depths the truck is seen at different levels of sharpness and we see that in plane 3 the truck is the sharpest and according to the AGMR got the maximum there.

1.4. Explanation of the Mask-R-CNN method [2]:

Mask R-CNN is an advanced deep learning algorithm for detecting and classifying object instances, introduced by Kaiming He, Georgia Gkioxari, Piotr Dollár and Ros Girshick in 2017. The algorithm extends previous algorithms on the subject, which dealt with object detection capabilities, by adding MASK prediction in each region of interest (ROI), parallel to what existed for classification by bounding boxes. This allows Mask R-CNN to not only detect objects within an image but also provide pixel-level masking for each detected object.

1.4.1 Components and architecture of Mask R-CNN

Network core:

-Convolutional Neural Network (CNN): A core network, mainly used for image recognition and processing, due to its ability to recognize patterns from the input image. This is essential for identifying objects and their exact location.

Region Proposal Network (RPN):

-RPN is a network Convolution which predicts object boundaries by learning from feature maps extracted from a base network. The network has a classifier that returns the probability of the region and regressor that returns the coordinates of bounding boxes that contain objects.

- the output of the RPN outputs a set of bounding boxes of candidate objects, along with the objects' scores indicating the likelihood that an object is in each box.

allocation ROI:

- Allocation ROI (region of interest) is a key innovation in Mask R-CNN that replaced the ROI Pooling layer used in Faster R-CNN. ROI assignment ensures accurate alignment of the extracted features with the input image, which is essential for MASK prediction accurately.

1.4.2 Mask-R-CNN capabilities:

-**Classification:** allocation a label for each area is suggested.

-**Bounding box regression:** Boundary the coordinates of The bounding box for each region is suggested.

-**MASK:** predict a binary MASK for each ROI, specifying the exact pixels belonging to the object within the bounding box.

1.4.3 Advantages of Mask-R-CNN:

1)**Accuracy test:** actional location ROI ensures accurate localization and detection, leading to high quality MASK object detection.

2)**flexibility:** can be easily expanded CNN-R-MASK for other tasks, such as human pose estimation.

3)**robustness:** The frame is resistant to variations in sizes, shapes and appearance of objects.

1.4.4 uses of Mask-R-CNN in the realization of the tracking and detection of distances:

-Mask R-CNN helps us discover objects and classifies them with it. Their classification is very important for tracking, since the most basic way to filter objects in tracking is according to the classification of the objects.

- The ROI and its bounding box are an important part of the algorithm for finding the Z-axis distances in the code (as explained in the introduction section).

- In addition the MASK of the model is used by us both visually and practically to calculate data such as the RGB ratios and irregularity- (will be detailed later).

1.4.5 Uses in different applications:

Mask R-CNN has been widely adopted in various fields due to its efficiency and accuracy in object detection and segmentation. Here are some uses from everyday life:

- **Autonomous driving:** Identification and segmentation of various objects (vehicles, pedestrians, traffic signs) in real time.
- **Medical imaging:** Segmentation of anatomical structures in medical images, such as tumors, organs and cells.
- **Augmented reality:** Classification Real-time objects to improve interaction with virtual objects in an augmented environment.
- **Robotics:** Allows robots to recognize and manipulate objects in busy environments.

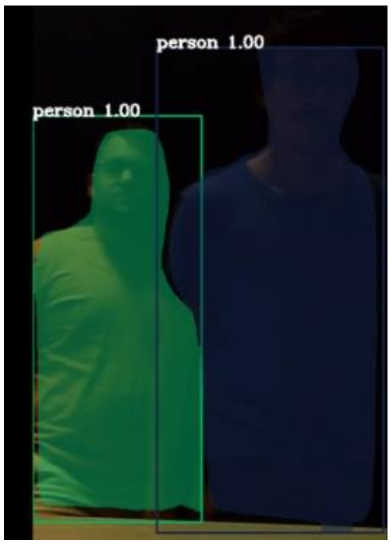


Figure 5. A picture from which you can see an application of Mask-R-CNN In the detection and segmentation phase. You can see the score received by each of the objects

2.Tracing of objects in a video

2.1. A brief background on tracking:

Classical tracking uses several properties, shape size properties, shape properties, brightness properties and position properties.

location characteristics	**Brightness characteristics**	**shape characteristics**	**shape size characteristics**
Center of gravity location C(x,y,z) of the shape	Different gray levels basically	Extent ratio	bone area
	The percentage of each color of the components of theR, G, B	compactness ratio	bone circumference
		irregularity	Blocking box area

These characteristics can be used by us in the tracking algorithm since many of them can differentiate between different objects and thus determine in cases where objects are close and it is impossible to rely on the location alone. However, many of these characteristics can change drastically between successive frames, so we cannot rely solely on one characteristic or a group of characteristics. Therefore, in our algorithm we will take into account the position of the object (in the three axes) and other characteristics from this list for the purpose of determining whether it is the same object or a new object between frames.

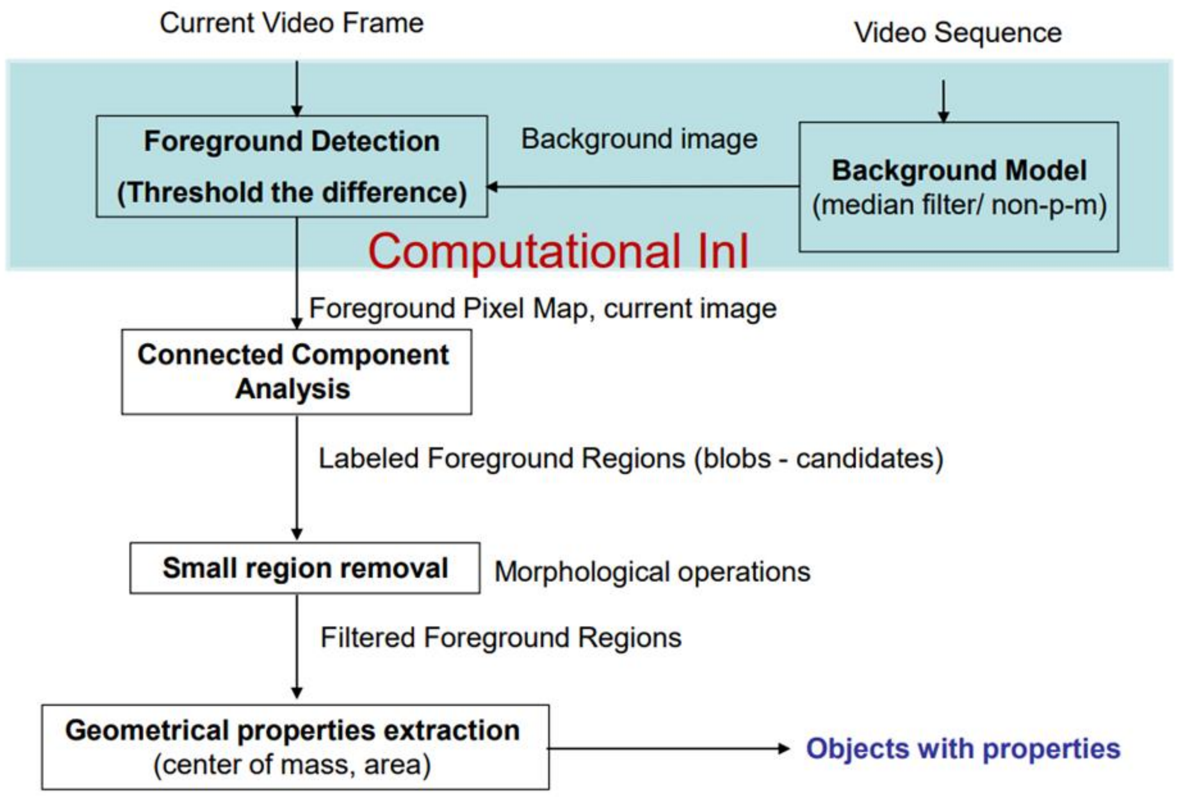
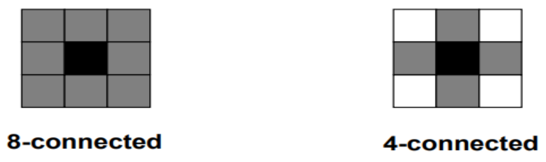


Figure 6. Diagram that describes the detection of the moving areas during tracking in noise and displacement conditions

Explanation of the step of Connected Component Analysis:

There are 2 types of Different connectivity -

-



According to the type of connectivity, it is possible to determine whether it is 2 different objects or actually one, in 4-connected If one of the sides has a pixel with the same bit then it is the same object and in 8-connected we will extend the test to see if there is the same pixel also at the corners and not just in the sides.

2.2. a feature based tracking process:

Detection of moving areas (bubbles)

Extracting the properties of each bubble

Traceability using the feature comparison process

This is the process we will do during the implementation of traceability in the code.

2.3. The idea of the tracking process:

The data we have:

List of objects that have been exiled - current frame t

List of exiled objects - next frame time t+1

The center of mass of an object in the current frame C_t^p

The center of mass of an object in the following frame C_{t+1}^i

threshold value is set so that if the objects are close the following condition is met:

$$Dist\left(C_t^p, C_{t+1}^i\right) < THRESH\_CM$$

Let's mark the sizes of the objects like this: S_t^p, S_{t+1}^i

threshold value is set greater than or equal to 1 for the similarity of the objects and they are similar if the following condition is met:

$$THRESH\_SIZE \geq \frac{S_t^p}{S_{t+1}^i} \geq \frac{1}{THRESH\_SIZE}$$

To determine whether 2 objects are an object p and object I - are the same object we will take into account that we will meet the two conditions listed above. And also for other parameters we will take into account each parameter with a different weight. It is possible that we will receive a match in which several objects meet the conditions we have defined and in such cases we will take the object that is the most suitable after considering the weight of each of the conditions.

3. Challenges in the tracking process:

- A situation where 2 objects will merge and appear as one object, a big challenge in these situations is to make the system understand that after the merging and the splitting of the objects which object corresponds to the object it was before the merge and to understand that these are the same objects (this challenge happens when these are objects of the same type, for example two people). The most convenient way to do this is to use the axis distances Z to identify who is who but sometimes it is not enough since the model of the distances becomes very difficult when two objects overlap in position with each other and drastically change their distances as a result. The model under these conditions behaves in an unreliable manner and therefore it is impossible to rely only on the distances in such a situation.
- There are situations where the model Mask-R-CNN does not function properly and produces unreliable results such as cases where the model stops recognizing a detected object for a few frames and then detects it again that it has already moved to another location or cases where the object receives a different classification in some frames and confuses the system (for example a detected car identified for a few frames as a truck and then back to being recognized as a car properly). In addition, sometimes the model detects objects that do not exist (such as a giraffe's leg that is detected as a human), but this malfunction has less effect on the tracking process.
- A situation where there are multiple objects of the same type, for example a video with many people, may cause the system to confuse the people if their characteristics are similar.

We will see an example of the problem of merging and splitting two objects and how, for example, this problem can be overcome. In this example a toy car passes a stationary car, the algorithm will have to understand that it is the same car that was before it passed the stationary car. In this case it can be seen that the contrast criterion of the two cars is different between the two cars and therefore if a higher weight is given to this criterion the system will be able to easily differentiate between the two vehicles.

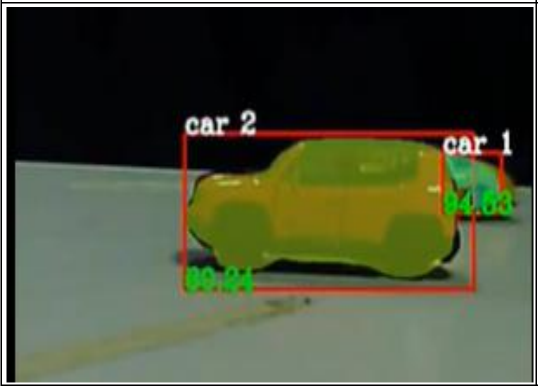
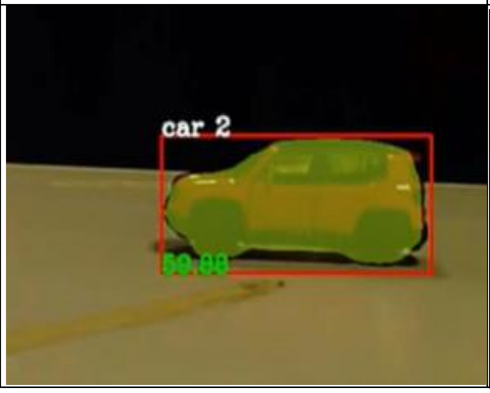
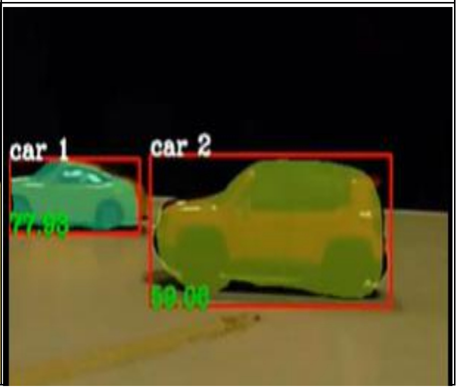
Here you can see the cars after the split. We will also notice that the cars keep the same label car 1 remained car 1 and-car 2 remains car 2.	Here the cars have merged and look like one bone	Here you can see the cars before the merger
		

Figure 7. An example of overcoming overlap between objects with the help of a parameter contrast.

3.1. Running the code:

during our experiment for running the code [1] we encountered some problems and decided to write some highlights:

- In the folder, the file that needs to be run is

3D object detection via 2D segmentation-based computational integral imaging.ipynb

-At first we tried to run in a work environment of collab but we didn't get along with it and moved to PyCharm for that we needed the professional version where you can install jupyter which is an extension that allows working with ipynb files

-the pycharm knows how to identify all the libraries required for installation and install them except 'pycocotools' which we had to install manually

- In addition, we tried to run in the Python version 3.9 but this version does not support all the libraries in the project and we had to downgrade to version 3.7.4

- During the running of the code we discovered that the code for finding the distances crashed if there is only one object in the frame or there are no objects at all (this problem has been fixed by us)

- Sometimes it was necessary to change the threshold parameters in the function to adapt them to each video in order for it to come out in a good way since we shot our videos and shot videos of toys at different distances and this affects many parameters, so we could not work for every video with the same characteristics.

4. Implementation of the algorithm:

The algorithm is implemented on the code we received for finding distances that was developed previously[REF1]. During the code the distances are calculated using the algorithm described at the beginning of the report. During this process, the objects are discover and identified for the purpose of finding the distances, during the work of the code in this part we added functions to calculate various parameters that will help us in the tracking to track and differentiate between the various objects.

Next we created a function called Trucker() that implements the algorithm and contains the actual tracking algorithm. Recall that the videos were taken with the help of the camera system described in the summary. The goal of our algorithm is to do tracking in a classical implementation where the innovation is the use of the Z-axis for this purpose. In addition to the Z-axis, the function uses additional properties to track and distinguish between objects, which are the distances of the objects according to each of the axes (X,Y), according to the RGB and according to the contrast. Additional characteristics of the objects were taken into account (such as irregularity, compactness, etc.) but after various experiments they were found to be less reliable and not good enough for tracking.

During the implementation, the most significant phase was the trial and error phase and the understanding of how much weight should be given to each parameter in determining the tracking and which parameters will be more significant to get better results. In addition, we had to understand which threshold values should be set for each characteristic, since they depend on the recorded videos.

Every time we made a comparison between the frames we took not only the next frame but several future frames to correct the errors that could arise along the way. For example, if in the next frame the algorithm identifies a certain object incorrectly, but in two further frames it correctly identifies it, then we will go with the correct identification.

What received the significant weight in determining the tracking is the distances (x,y,z) which are the parameters that can be relied on the most but we could not rely only on them since in overlapping situations or situations where objects are adjacent, the distances can be confusing and therefore we relied on the RGB and contrast in addition to decide between the objects in these cases.

The cameras are of medium quality and change colors during the video, which affects the quality and reliability of the RGB and in addition on the quality of the contrast results, therefore many times we received wrong results in tracking in situations of overlapping objects.

Each of these parameters was given a different weight in determining whether the object in the current frame is the same object in the previous frame according to a certain threshold value and this is of course reflected in the writing of the code.

For different videos, we shot different objects and each parameter of the various relevant parameters we test works differently. That's why for different videos we will do a different weighting and weighting of the parameters, that is, for each video we did a process of trial and error since we don't currently have a system that knows how to identify on its own what is the right weight for each video in order to perform the tracking in the best way for each video. Because when we worked with uniform values for all situations we dont got a results that satisfies us.

The algorithm returns us the objects framed in a frame and marked according to the model we received and writes above the frame whether it is person 1 or person 2, car 1 or car 2 and continues to do this consistently until the end of the video for that object. When in the bottom of the frame he will write us the distance in the Z axis according to the video.

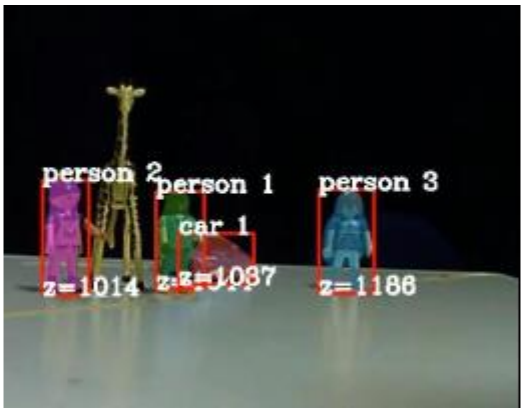


Figure 8. An illustration in which you can see the tracking of the people and their distances. We will notice that each object is marked as Person 1, person 2, person 3, car 1

4.1. A brief description of the algorithm:

In the first stage of running the algorithm, the algorithm classifies the objects in the first frame and gives them names (for example if there are three people it will give them the names person1, person2, person3).

The algorithm loops through all the frames and for every frame, the code looks for a match in the next frame and if not it does the same for a certain number of frames ahead.

for object a in frame x at first the code looks for the object closest to the object in frame x+1 according to a distance in the three axes that is classified as the same type (i.e. for a human we will only search for humans in the following frames), this distance should be close i.e. smaller than distance thresh we defined. If there is only one object close to object a, we choose this object. Otherwise, to decide, we will look for a match according to the contrast , RGB and the Z distance. Among the objects that meet the threshold condition of these parameters (in terms of distance between the parameters) we will weigh the distances and all the parameters of the objects and object a according to predetermined weights and choose the object that is closest in value to object a if there is one. If no match was found for object a in frame x+1, we will try the same process in frames x+2 to x+m where m is the number of the next frames we will look for if the object exists in them (we set m=5 in our code). If we do not find a match, we will decide that this object will no longer appear in the video. If we found a matching object we mark it as b.

If we found an object b matched We will check if another object from the same frame x or from a previous frame x-i, decided to classify it as the same object. If it has already been classified by another object, we will check which object is more suitable for object b according to the conditions detailed earlier. The object that is less suitable returns to the list of objects that are looking for a match and object b is dropped from its options for matching.

To summarize the operation of the tracking function:

The function gives us a powerful method for tracking objects across multiple frames in a row in short videos. The method ensures that each object is uniquely identified and consistently tagged throughout the video. This is done by considering the coordinate values in space in addition to the coordinate Z, contrast and RGB values. The function matches labels as accurately as possible to objects across frames, even if they temporarily disappear or move significantly. The iterative and hierarchical approach to checking successive frames ensures reliable tracking, making the function a valuable tool for video analysis and object tracking applications.

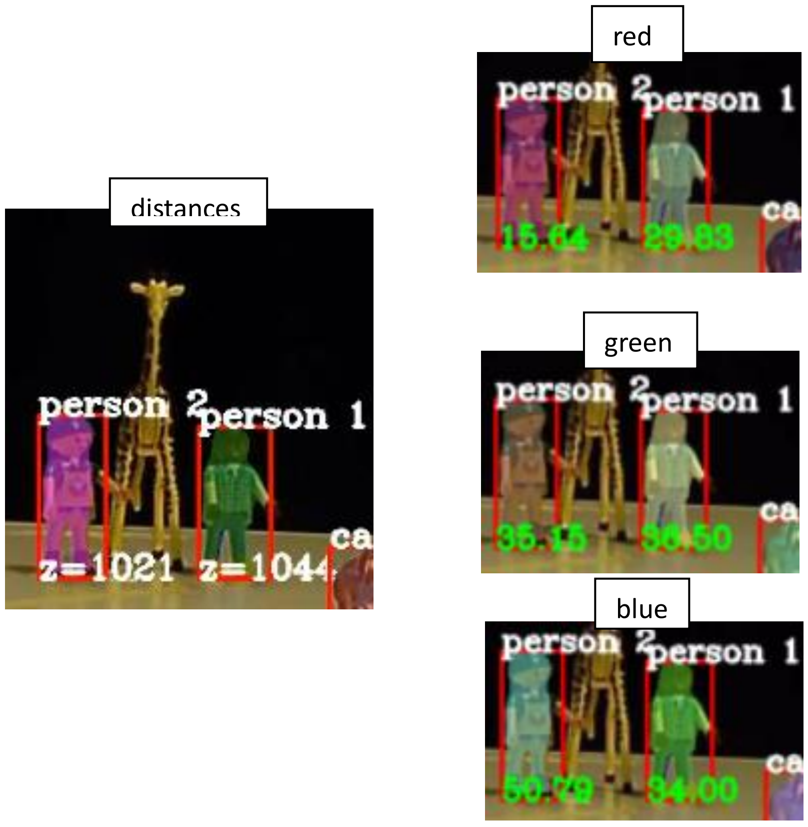


Figure 9. Example when two objects of the same type are in proximity and it’s difficult to distinguish between them. in this example we can distinguish between them by looking at the RGB values to determine who is who.

We will see the process of classifying the objects in the diagram:

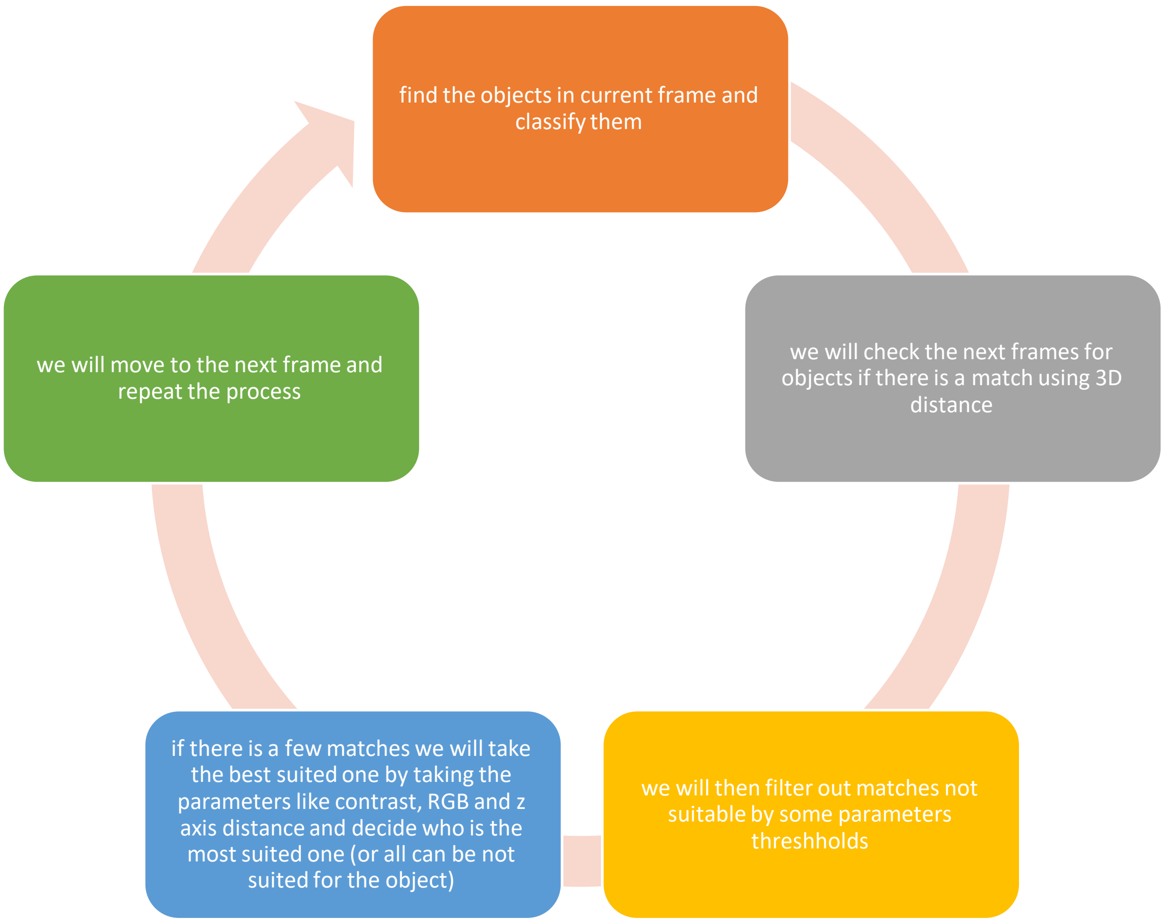

find the objects in current frame and classify them
we will check the next frames for objects if there is a match using 3D distance
we will then filter out matches not suitable by some parameters threshholds
if there is a few matches we will take the best suited one by taking the parameters like contrast, RGB and z axis distance and decide who is the most suited one (or all can be not suited for the object)
we will move to the next frame and repeat the process

Figure 10. The tracking process diagram

5. results and improvements:

- The tracking works well for most videos and performs tracking in high percentages and the algorithm is able to distinguish between close objects and objects that merge and diverge between them. But the tracking is not perfect and sometimes there are glitches where the tracking of a certain object is lost, meaning that in one frame it changes the classification of the object to something else.

-We measure the success of a video by the percentage of frames where the tracking managed to work perfectly meaning the frames which all objects are detected and given the right classification and names (i.e. person1, person2, car1…) of the total numbers of frames.

$$Tracking\ success = \frac{number\ of\ perfect\ frames}{total\ number\ of\ frames}$$

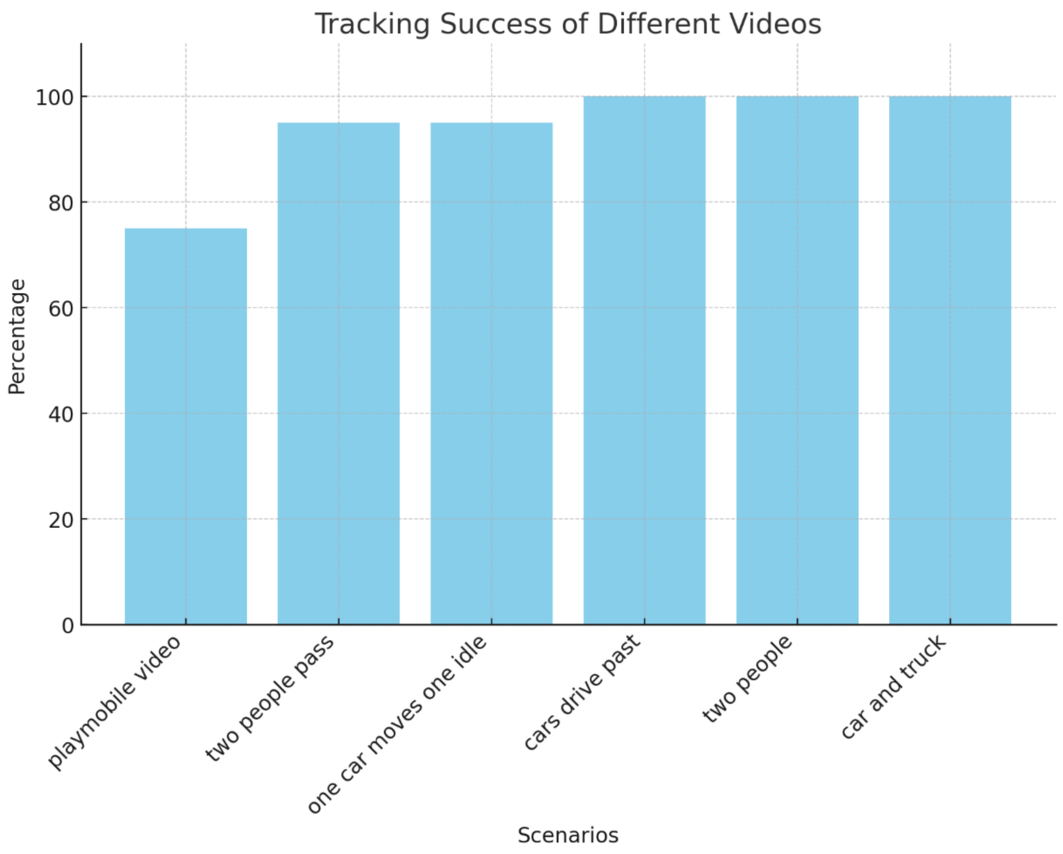


Figure 11. Tracking success for different videos

-The frames were checked visually by us to verify whether they were successful or not we measured the results on many different videos with challenging alignment situations and in most videos the success rate is between 75 to 100 precent of success with the majority above 95 precent.

-The majority faults of the tracking if not all of them were due to the failure of the Mask-R-CNN to track objects for certain frames. However, the tracker managed to track identify those same objects in the following frames.

- In order to get the best results for each video we will have to slightly change the weight of the parameters and the threshold values in order to perform the tracking in the best way since with fixed values the algorithm does not produce ideal results and also if we want to improve the algorithm in the future we would like to add an automatic way in the algorithm To determine the weights and threshold values of the parameters.

- It is possible and desirable to upgrade the array of cameras to get better results and that the parameters for which we decide on the object will be more reliable than the necessity of the upgrade. In addition, it is possible to improve the algorithm for finding the distances, which will bring slightly more stable results, especially in situations of overlap between objects.

A few of the results can be seen here:

https://www.youtube.com/@Trackingobjectsin3DYM some examples:

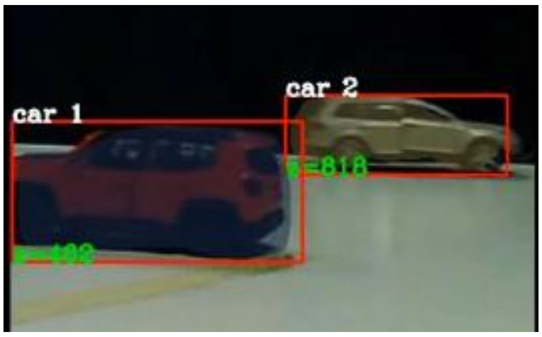


Figure 12. We can see 2 objects passing each other with a clear classification that these are two different objects and see their distances.

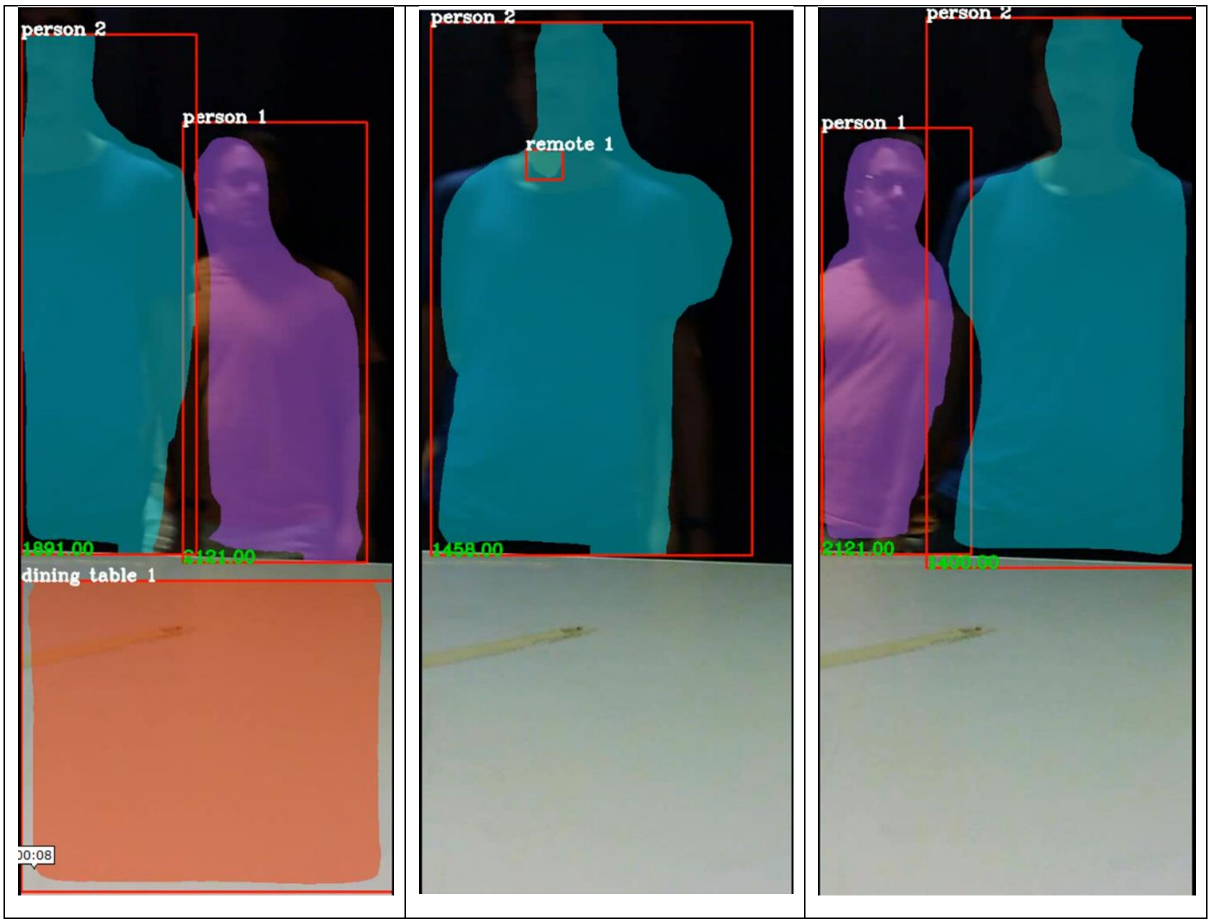


Figure 13. When passing one over the other the system was still able to track each of us correctly despite the difficulties of the distances that were not stable in this situation and the identification of the excess objects

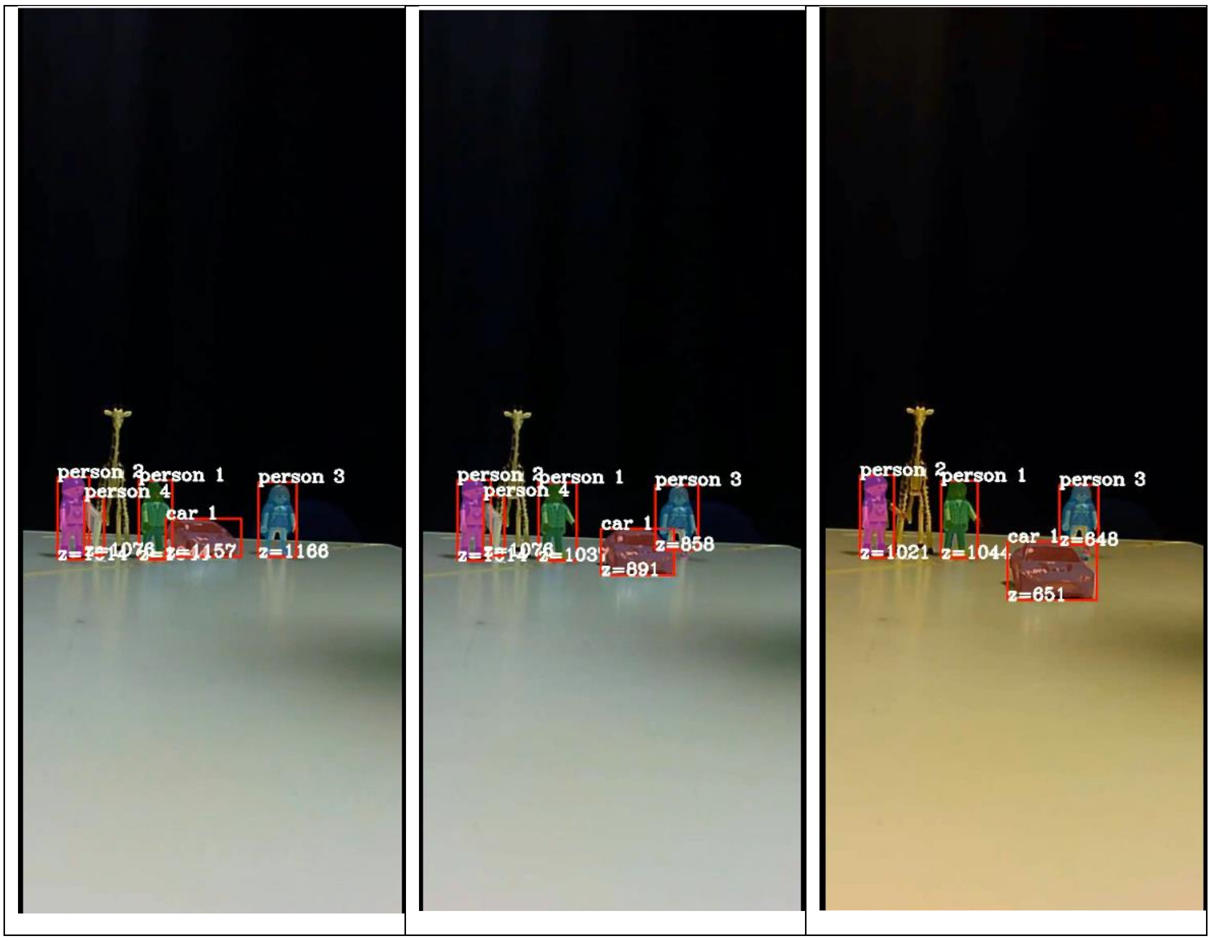


Figure 14. Here we can see many different objects that the model manages to follow accurately (from right to left: the beginning of the video, the middle of the video and the end of the video) despite the jumps in distances. It can be seen that the model recognizes the giraffe toy leg as another human from the middle of the video but this does not interfere with the tracking. (In addition, the model was not trained to identify a giraffe therefore does not know how to identify this toy)

6. Conclusions:

The project successfully demonstrates the application of object tracking in a classical approach while incorporating an innovative use of the Z axis to follow the objects in the 3D space. Through the development and execution of the code we introduced moving object tracking using Mask-R-CNN and we emphasized the importance of the Z axis and other parameters for tracking accuracy. However, the limitations of the current model in determining distances along the Z axis indicate the need for further refinement of calculation at a distance.

In the project we combined the theoretical knowledge we learned from the article with actual coding, this makes it possible to apply classical tracking methods and realize them in the real world. The findings highlight the potential in three-dimensional tracking for future research and development in this area, especially in combination with artificial intelligence and real-time data processing techniques to address future challenges in object tracking.

By improving the accuracy of the current model and exploring additional tracking parameters, this work lays a foundation for advances in video analysis and object tracking applications. The insights gained from this project can contribute to future innovations and improvements in the field.

7: References

[1] – M. Kadosh, Y. Yitzhaky, “3D Object Detection via 2D Segmentation-Based Computational Integral Imaging Applied to a Real Video”, Sensors 23 (9), 2023. https://doi.org/10.3390/s23094191

[2] - He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 2961-2969. 2017.

[3] – D. Aloni; Y. Yitzhaky; Automatic 3D Object Localization and Isolation Using Computational Integral Imaging. Appl. Opt. 2015,54, 6717.

[4]- Y. Qian, H. Shi, M. Tian, R. Yang and Y. Duan, "Multiple Object Tracking for Similar, Monotonic Targets," 2020 10th Institute of Electrical and Electronics Engineers International Conference on Cyber Technology in Automation , Control, and Intelligent Systems (CYBER), Xi'an, China, 2020, pp. 360-363

[5] - WA Khan, DR Pant, B. Adhikari and R. Manandhar, "3D object tracking using disparity map," 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 2017, pp. 1015-1018.

[6] - L. Zhao, M. Wang and Y. Yue, "SCMOT: Improving 3D Multi-Object Tracking via Semantic Inference and Confidence Optimization," 2024 36th Chinese Control and Decision Conference (CCDC), Xi'an, China, 2024, pp.

[7] - A. Mangawati, Mohana, M. Leesan and HVR Aradhya, "Object Tracking Algorithms for Video Surveillance Applications," 2018 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2018, pp. 0667-0671.

[8] - W. Feng, C. Yu, F. Meng and A. You, "Pedestrian Tracking Algorithm for Siamese Networks Based on Improved Attention Mechanism and MASK R-CNN Detection," 2024 5th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT), Nanjing, China, 2024, pp. 752-756.