automation

MDPI

Article

Vision-Based Object Manipulation for Activities of Daily Living Assistance Using Assistive Robot

Md Tanzil Shahria [1,*,†], Jawhar Ghommam [2], Raouf Fareh [3] and Mohammad Habibur Rahman [1,4]

[1] Computer Science, University of Wisconsin-Milwaukee, Milwaukee, WI 53211, USA; rahmanmh@uwm.edu
[2] Electrical and Computer Engineering, Sultan Qaboos University, Muscat 123, Oman; jawher@squ.edu.om
[3] Electrical and Computer Engineering, University of Sharjah, University City, Sharjah 27272, United Arab Emirates; rfareh@sharjah.ac.ae
[4] Mechanical Engineering, University of Wisconsin-Milwaukee, Milwaukee, WI 53211, USA
* Correspondence: mshahria@uwm.edu
† Current address: Biorobotics Laboratory, University of Wisconsin-Milwaukee, 115 East Reindl Way, USR 281, Milwaukee, WI 53212, USA.

Abstract: The increasing prevalence of upper and lower extremity (ULE) functional deficiencies presents a significant challenge, as it restricts individuals' ability to perform daily tasks independently. Robotic devices are emerging as assistive devices to assist individuals with limited ULE functionalities in activities of daily living (ADLs). While assistive manipulators are available, manual control through traditional methods like joysticks can be cumbersome, particularly for individuals with severe hand impairments and vision limitations. Therefore, autonomous/semi-autonomous control of a robotic assistive device to perform any ADL task is open to research. This study addresses the necessity of fostering independence in ADLs by proposing a creative approach. We present a vision-based control system for a six-degrees-of-freedom (DoF) robotic manipulator designed for semi-autonomous "pick-and-place" tasks, one of the most common activities among ADLs. Our approach involves selecting and training a deep-learning-based object detection model with a dataset of 47 ADL objects, forming the base for a 3D ADL object localization algorithm. The proposed vision-based control system integrates this localization technique to identify and manipulate ADL objects (e.g., apples, oranges, capsicums, and cups) in real time, returning them to specific locations to complete the "pick-and-place" task. Experimental validation involving an xArm6 (six DoF) robot from UFACTORY in diverse settings demonstrates the system's adaptability and effectiveness, achieving an overall 72.9% success rate in detecting, localizing, and executing ADL tasks. This research contributes to the growing field of autonomous assistive devices, enhancing independence for individuals with functional impairments.

Keywords: activities of daily living (ADLs); assistive robot; localization; pre-trained deep learning model; robotic assistance; vision-based object manipulation

Citation: Shahria, M.T.; Ghommam, J.; Fareh, R.; Rahman, M.H. Vision-Based Object Manipulation for Activities of Daily Living Assistance Using Assistive Robot. *Automation* **2024**, *5*, 68–89. https://doi.org/10.3390/automation5020006

Academic Editor: Felipe Martins

Received: 1 March 2024
Revised: 8 April 2024
Accepted: 11 April 2024
Published: 15 April 2024

1. Introduction

Functional impairments of the upper and lower extremities (ULEs) are common due to stroke, spinal cord injury (SCI), cerebral palsy (CP), multiple sclerosis (MS), amyotrophic lateral sclerosis (ALS), trauma, occupational injuries, and geriatric disorders [1,2]. Stroke accounts for 34% of all cases of ULE dysfunctions (ULEDs) [3], with SCI (27%), MS (19%), and CP (8%) following closely behind. Stroke affects more than 15 million people each year worldwide [4]. ULEDs affect roughly 50–80% of stroke survivors in the acute phase and 40–50% in the chronic phase [5]. Also, arm and hand functions decline dramatically with age [6]. Over the last decades, the number of individuals with ULEDs has increased alarmingly. Approximately 61 million adults in the United States live with a disability [7]. Over 6.8 million Americans use assistive devices to help them, 1.7 million are wheelchair users, and almost one-third of mobility device users need assistance from another person in

one or more of the activities of daily living (ADLs) [8]. The inability to complete necessary daily tasks may result in unsafe situations and low quality of life.

The human upper extremity (UE) is the most highly developed anatomical structure because of its usefulness in manipulating and communicating with various objects. UE mobility is essential to the overall quality of life because most ADL tasks depend on the UE in some form, whether entirely or partially. The loss of UE function severely impairs the affected person's ability to engage in daily activities independently [9]. The inability to use one's UE typically has far-reaching consequences for a person's everyday life and independence in the workplace and at home. The problem is further compounded by the constantly growing number of such cases.

Existing solutions involve caregivers and/or assistive devices to assist with specific tasks. Many organizations offer disabled-adapted facilities to support them. Yet, it has been observed that disabled-adapted facilities can restrict their mobility [10], even though such facilities are rarely utilized. Therefore, there is an urgent need for research on ADL assistance focusing on individuals with ULEDs to enhance the self-sufficiency of these individuals and, in turn, reduce the caregiving responsibilities of their families.

Assisting individuals with ADLs is an essential healthcare service that supports independent living for people with disabilities, chronic illnesses, or aging-related limitations. However, the increasing demand for such assistance exceeds the availability of caregivers. So, these individuals have to wait for the availability of the caregiver, even for the essential ADL tasks. Approximately 41 million caregivers are estimated to offer over 34 billion hours of valuable care to their families and loved ones each year, with an estimated worth surpassing USD 470 billion [11]. Recent advancements in robotics and computer vision technologies have opened up new opportunities for developing intelligent systems that can provide personalized, efficient, and reliable ADL assistance. These technologies can help such individuals to regain their independence by minimizing their dependence on caregivers.

A few robotic manipulators available on the market can assist the user, yet controlling the robot sometimes becomes challenging and complex, especially for individuals with severe limitations of hand functions and distance vision impairment. In our preliminary work, we designed a minimum viable product (MVP) prototype of a multifunctional robotic assistive arm (mR2A) (supported by the National Institute on Disability, Independent Living, and Rehabilitation Research, Grant # 90DPGE0018-01-00). The participants' feedback suggests that there is still room for more significant advancements toward the practical implementation of robotic assistive devices. For example, operating the robot can be difficult, especially for those with severe hand function and distance vision impairment. Thus, some form of automation inside the system is necessary to help the user utilize the system properly and reduce the burden of care for the family.

This research aims to design, develop, and evaluate a vision-based semi-autonomous ADL assistance system using a commercially available assistive robot to enhance the quality of life by providing ADL assistance to wheelchair users with the "pick-and-place" task. Furthermore, this study aims to improve the user's well-being. The proposed system uses computer vision, deep learning, and localization techniques to perform "pick-and-place" tasks, the most prevalent activity among ADLs [12]. The system will enable users to control their actions while providing semi-autonomous assistance as needed.

The primary contribution of this research is developing a vision-based robot manipulation system to perform the "pick-and-place" ADL task semi-autonomously. Naturally, it is challenging for people with disabilities to pick an object from an upper shelf. They often seek help from a caregiver or family member to perform this ADL task, which is one of the most common. The system integrates advanced techniques, such as deep-learning-based object detection and localization algorithms, with a control algorithm to enable a robotic manipulator to perform the task for them semi-autonomously. The experimental results demonstrate the effectiveness of the vision-based robot manipulation system in performing the ADL task and the potential for further research in this area. This research is the proof of concept that assistive robots can be used to perform ADL tasks autonomously. Although

the long-term goal of this research is to automate various ADL tasks, this research focuses explicitly on the semi-autonomous performance of the "pick-and-place" ADL task. In future research, we will expand the system's capabilities to perform additional ADL tasks autonomously in various workspaces. The findings from this research will guide us in enhancing the system's performance and applicability in real life.

The rest of this paper is organized as follows: Section 2 presents a brief literature review on vision-guided ADL assistance and a general review of vision-based robotic applications. Section 3 describes developing a deep-learning-based ADL object detection model, including selecting, fine-tuning, and evaluating a pre-trained deep learning model. Section 4 presents an algorithm for accurately localizing ADL objects in a 3D environment, leveraging the deep learning model from Section 3 and employing computer vision techniques and depth estimation. In Section 5, the design of the vision-based robot manipulation system for ADL tasks is illustrated, encompassing the integration of detection and localization algorithms with a robotic manipulator and the development of control algorithms enabling the robot to perform ADL tasks. Section 6 showcases the experimental results of the vision-based robot manipulation system, including details on the experimental setup, results, and analysis of the system's performance in executing the "pick-and-place" ADL task. Finally, Section 7 summarizes the research, discussing the contributions and limitations of the present work and identifying potential areas for future research in vision-based robot manipulation.

2. Literature Review

This section briefly reviews recent developments in vision-guided ADL assistance using assistive robots and a comprehensive analysis of the state of the art in the development of vision-based robotic applications.

2.1. Vision-Guided ADL Assistance

Assistive robots have gained significant attention in recent years as a potentially transformative technology for improving the lives of individuals with disabilities and older adults [13]. These robots offer physical and cognitive assistance, particularly for ADLs, where individuals face challenges in independent performance [14,15]. Among the various methods explored for ADL-assisting robots, vision-guided assistance has emerged as one of the most promising approaches [14–16]. Vision-guided ADL assistance leverages computer vision algorithms and sensors to enable robots to perceive and interact with their environment [17]. This technique holds immense potential for accurately identifying and locating objects in real-world environments, which is critical for many ADL tasks.

Recent advancements in vision-guided ADL assistance using assistive robots demonstrate notable progress. The utilization of deep learning algorithms for object detection and recognition has emerged as a significant development [18]. These algorithms, capable of learning from extensive datasets, enable robots to detect and identify objects with higher precision and reliability [19]. They have successfully contributed to various ADL tasks, including object detection for meal preparation and object recognition for medication management.

Additionally, there is a growing interest in developing personalized and adaptable assistive robots. Machine learning algorithms that adapt to user behavior and preferences enable personalization [19,20]. For instance, a robot can learn a user's preferred method of meal preparation and adjust its assistance accordingly. Adaptable robots can also accommodate changes in a user's abilities over time [21]. These advancements hold immense potential for the future of vision-guided ADL assistance.

While vision-guided ADL assistance offers significant promise, several challenges persist [22]. Developing robust algorithms for object detection, tracking, and recognition is a major challenge, along with ensuring the safety and reliability of robot actions, particularly in tasks involving physical interaction with users [22,23]. Moreover, creating more personalized and user-friendly interfaces is essential for effective human–robot interaction.

Addressing the challenges requires collaborative efforts among researchers, engineers, designers, and users. These advancements will contribute to developing advanced assistive robots that provide practical and personalized ADL assistance, reducing the dependence on caregivers and enhancing individuals' independence in their daily lives. By focusing on autonomous control methods and addressing the challenges inherent in this approach, more capable and user-friendly robotic systems can be created for ADL support.

2.2. Vision-Based Robotic Applications

Recent advancements in robotics have significantly enhanced the versatility and capability of robots for real-world tasks. Vision-based techniques, in particular, have become central to robotic manipulation, offering rich sensory feedback and overcoming various challenges in control and application domains, such as object handling, navigation, and interaction [24]. Learning-based control theories have emerged as front-runners in this area due to their ability to process extensive datasets and adaptively improve, thereby refining robots' manipulation accuracy and efficiency in dynamic settings. These methods leverage an array of cameras—RGB, depth, stereo, and monocular—to perceive the environment accurately. However, researchers still grapple with challenges related to reflected patterns, drift, accumulation error, low spatial resolution, line-of-sight obstruction, and ambient light saturation.

Recent advancements in vision-based robotic manipulation have focused on overcoming challenges such as occlusion and complex environments. Yin et al. [25] developed an RGB-D-based approach that enhances robotic grasping within fusion application environments by integrating instance segmentation with clustering and planar extraction, thus improving stability and adaptability in uncertain conditions. Similarly, Yu et al. [26], through their innovative use of a single shot multibox detector (SSD)-based detector, an image inpainting and recognition network, and deep grasping guidance network, proposed a novel method to address occlusions in robotic grasping. This method leverages image inpainting within the detection process to significantly enhance object detection under occluded scenarios, facilitating more efficient determination of optimal grasping poses.

Building on these concepts, James and Davison [27] introduced Q-Attention, an attention-driven robotic manipulation (ARM) framework that efficiently learns from demonstrations to perform sparsely rewarded tasks. ARM utilizes a Q-Attention module to focus on relevant parts of the scene, simplifying complex manipulation tasks into manageable steps and showing impressive adaptability across different tasks. Additionally, Zhang et al. [28] presented a dual neural network controller approach that enables robots to grasp objects in visually complex environments. Their method demonstrates improved grasping performance and robustness in environments by combining visual guidance with a dual network system that separately processes object recognition and grasp stability.

These studies collectively illustrate the significant strides made in addressing the inherent challenges of robotic manipulation. Despite the significant progress achieved by vision-based object manipulation systems, challenges related to sensor limitations and environmental conditions persist. Further research studies are essential to overcome these challenges and advance the development of more advanced and reliable vision-based object manipulation systems.

2.3. Findings from the Review

This literature review provides crucial insights into vision-based robot manipulation. It emphasizes the extensive research devoted to enhancing robot manipulation through vision-based methodologies, focusing on integrating cameras and sensors to provide robots with enhanced capabilities. In addition, the review highlights the variety of approaches in vision-based robot manipulation, such as object detection, motion planning, control, and machine learning techniques, each of which has distinct benefits and limitations based on the application. Notably, the choice of camera type is crucial, with depth cameras proving especially useful for human-interface systems due to their capacity to acquire

three-dimensional environmental data, facilitating enhanced robot navigation. Based on a comprehensive understanding of prior research methods, these findings provide invaluable guidance for developing a robust and practical vision-based robot manipulation system.

2.4. Potential Research Gap and Possible Approach

The review identifies specific research gaps in vision-based robot manipulation that demand attention. The need for algorithms capable of handling complex tasks in dynamic environments and the development of vision-based manipulation techniques for autonomous or semi-autonomous object manipulation are examples of these gaps. To address these issues, a promising strategy combines vision-based detection and recognition models with 3D localization algorithms that utilize depth camera data for real-time object localization. This information can then facilitate semi-autonomous object manipulation, which is especially helpful for users who struggle with manual robot control. The proposed strategy encompasses theoretical exploration, algorithm refinement, and experimental validation, offering the potential to expand robotic capabilities into a broader range of tasks within complex environments.

3. Deep-Learning-Based Detection Model

In recent years, deep-learning-based detection models have been gaining popularity due to their ability to accurately and efficiently identify objects in images, videos, and other forms of data. Deep learning is a subfield of machine learning that models and processes complex data structures using complex neural networks [29]. At its heart is the artificial neuron, producing an output based on the equation

$$y = \sigma(wx + b) \tag{1}$$

where w, x, b, and σ represent the weight vector, the input vector, the bias, and an activation function, respectively. Deep learning architectures can capture complex hierarchical patterns in data by stacking layers of these neurons, enabling advances in fields ranging from computer vision to natural language processing.

Deep learning models are typically coupled with convolutional neural networks (CNNs) for tasks like object detection from image data. CNNs are a subset of deep neural networks explicitly designed to process data structures that appear like grids, most notably images. Convolutional layers, which use discrete convolution operations to capture local patterns, are vital components of CNNs [30]. Additionally, pooling layers reduce the size of the space while keeping essential features, and fully connected layers allow for reasoning across all datasets. CNNs' weight-sharing feature reduces the number of parameters, improving generalization and preventing overfitting. By utilizing these structures, CNNs have transformed image recognition and classification and have come to be essential in computer vision tasks.

Detection models based on deep learning have found applications in diverse fields, such as object detection, face recognition, anomaly detection, and more [31]. They have achieved state-of-the-art performance on various benchmark datasets, making them a powerful research tool. However, these models require huge amounts of labeled data and computational resources, frequently unavailable or impractical for many real-world applications.

Utilizing a pre-trained model for any detection task is gaining popularity due to its numerous advantages [32]. Pre-trained models have already undergone training on extensive datasets, enabling them to recognize and classify a wide array of features and patterns effectively. This prior learning facilitates faster and more efficient training of the model for specific applications. Pre-trained models are often equipped with advanced features like data augmentation and regularization techniques, developed and evaluated by experts. Overall, using a pre-trained model can save time, improve accuracy, and provide access to state-of-the-art models without requiring extensive training and computational resources.

This section presents the selection and training of a deep-learning-based pre-trained detection model for ADL object detection. Following the model selection, we conducted fine-tuning using a custom dataset comprising images of various ADL objects. The primary objective was to evaluate the performance of the pre-trained detection model and test its accuracy and speed on the custom ADL dataset.

3.1. Dataset

Deep learning algorithms are designed to learn from large amounts of labeled data, allowing them to extract patterns and features that can be used to make accurate predictions on new, unseen data. The quality and quantity of the dataset used to train a deep learning model can substantially affect its performance [33], as a poorly labeled or biased dataset can lead to inaccurate or unfair predictions. A diverse dataset with numerous examples and variations is essential to ensure the model's ability to generalize to new data.

Dataset generation for deep learning models requires the creation of a labeled dataset that can be used for training and testing the model. An alternative approach involves the utilization of open datasets. Open datasets are accessible to the public and can be utilized for research, educational, or other purposes [34]. ImageNet [35], COCO [36], and PASCAL VOC [37] are examples of well-known open datasets utilized for deep learning.

COCO (Common Objects in Context) Dataset

In this research, we have collected data from the COCO (Common Objects in Context) dataset [36], a large-scale image recognition, segmentation, and captioning dataset that is usually utilized for object detection and image segmentation tasks. The dataset comprises more than 328,000 images and 2.5 million labeled object instances, encompassing various object categories such as humans, animals, automobiles, and household commodities. The dataset is intended to represent real-world scenarios by capturing images from various sources and contexts. The provided images exhibit a range of diversity and present challenging scenarios, featuring complex compositions with multiple objects, occlusions, and backgrounds that exhibit variability.

Choosing the COCO dataset for ADL object detection offers several advantages, including high-quality annotations, a large number of object categories, and a diverse set of images that can improve the model's robustness and precision. As this research focuses on manipulating ADL objects, 47 classes (as shown in Table 1) were extracted from the COCO dataset and used to train the ADL detection model. In our preliminary work, we identified that to provide ADL assistance robots need to manipulate 79 ADL objects [38]. These 47 classes were selected by intersecting the 79 essential ADL objects and 90 classes available in the COCO dataset. In total, 34,627 images were utilized to train the detection model, while 1473 were used to validate it. To mitigate overfitting, we ensured that the images in the testing set were not present in the training set. Overall, our custom dataset allowed us to evaluate the performance of our pre-trained detection model on a diverse set of images and objects.

Table 1. List of 47 ADL objects identified from the COCO dataset.

1	Apple	2	Backpack	3	Banana
4	Bottle	5	Bench	6	Bed
7	Bowl	8	Book	9	Broccoli
10	Chair	11	Cake	12	Cell Phone
13	Clock	14	Couch	15	Cup
16	Carrot	17	Dining Table	18	Donuts
19	Fork	20	Handbag	21	Hot dog
22	Hair Drier	23	Keyboard	24	Knife

Table 1. *Cont.*

25	Laptop	26	Microwave	27	Mouse
28	Orange	29	Oven	30	Pizza
31	Remote	32	Refrigerator	33	Sandwich
34	Spoon	35	Scissors	36	Sports Ball
37	Suitcase	38	Sink	39	Toothbrush
40	Tie	41	Teddy Bear	42	Toaster
43	Toilet	44	TV	45	Umbrella
46	Vase	47	Wine Glass		

3.2. Model Selection and Training

Selecting a pre-trained model for a specific task requires evaluating the performance and suitability of various models based on accuracy, speed, resource requirements, and task-specific requirements. Several popular pre-trained object detection models, including SSD MobileNet [39], YOLOv4 [40], YOLOv5 [41], and YOLOv7 [42], were considered for this specific task. SSD MobileNet's efficiency makes it an excellent option for embedded devices. Similarly, YOLO4 has a good accuracy, employing a larger backbone network to achieve state-of-the-art performance. However, both models are slightly slower and more resource-intensive than other models. YOLOv7 is a recent innovation that employs a larger and more complex backbone network to achieve even greater precision. Yet, due to its computational requirements, it may not be suitable for real-time or embedded applications. While each model has advantages and drawbacks, YOLOv5 offers a good balance of speed and accuracy, making it a practical and versatile option for various object detection tasks.

YOLOv5

YOLOv5 [41], a popular object detection algorithm, is widely utilized for object detection, employing a single-stage architecture and anchors to identify objects within an image. It includes new optimizations and features that boost performance, making it an excellent option for various applications. The utilization of CSPDarknet, a more proficient backbone network, enables expedited inference times while concurrently attaining high precision. Additionally, YOLOv5 is pre-trained using the COCO dataset [36], which features a variety of object categories and is intended to reflect real-world scenarios. Overall, YOLOv5 is a well-balanced and effective real-time pre-trained detection model.

YOLOv5 has several variations [43], including YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. Each variation differs in size and complexity. YOLOv5s stands out as the most compact and fastest variant, tailored for real-time detection on mobile and embedded devices. With fewer parameters and layers, YOLOv5s is well suited for devices with limited processing power. Despite being small, YOLOv5s outperforms its predecessors and other cutting-edge object detection algorithms in terms of accuracy. The 640×640 input resolution and real-time operation of the YOLOv5s model make it suitable for real-time detection in video streams.

YOLOv5m, YOLOv5l, and YOLOv5x, on the other hand, are larger and more complex models designed for greater accuracy but with a tradeoff in speed and computational resources. These models have more parameters, deeper layers, and higher input resolutions than YOLOv5s, making them more precise and slower. They are appropriate for applications requiring high precision.

Based on these advantages, we selected the YOLOv5s model as the base architecture for our real-time ADL object detection tasks, as it balances speed, accuracy, and computational resources. We fine-tuned the pre-trained model using our custom dataset. Fine-tuning allows us to adapt the pre-trained model to our ADL object detection task in images. While training the model, a learning rate of 0.01 and a batch size of 32 were used. We monitored the loss and accuracy metrics during training to ensure the model learned effectively.

Overall, our fine-tuned, pre-trained detection model allowed us to use the power of deep learning while reducing the amount of labeled data needed for training.

3.3. Results

We evaluated the performance of our pre-trained YOLOv5s detection model on our custom dataset. We used the mean average precision (mAP) metric to measure the model's performance. The inference time of our model is competitive, with an average of 5.1 ms, and the average pre-processing per image is 0.2 ms. Here, the training and validation loss for each epoch of the trained model is illustrated in Figure 1, along with the mean average precision (mAP). We can see that the loss is higher initially but converges after 50 epochs.

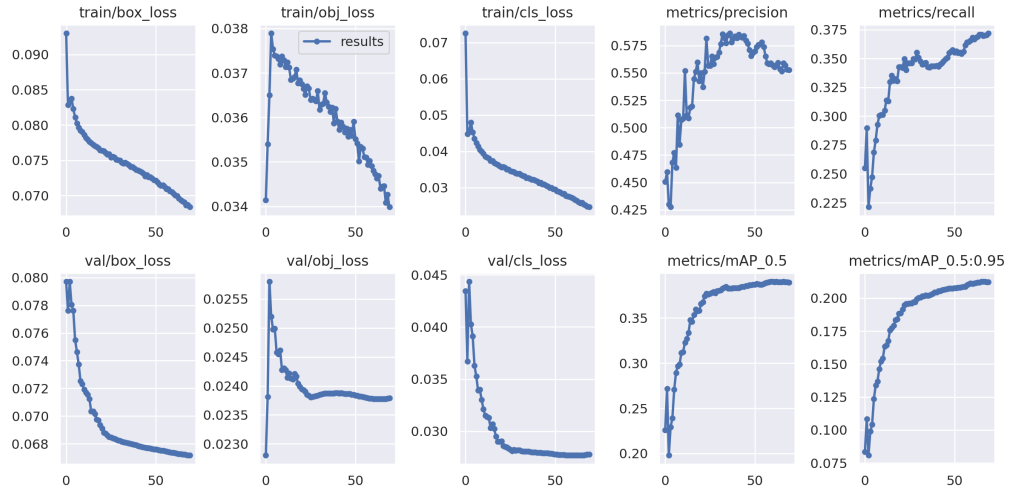

Figure 1. Train and validation loss of the model for each epoch.

Overall, the pre-trained detection model achieved an mAP@0.5 (mean average precision calculated at an intersection over union (IoU) threshold of 0.5) of 0.39 on our testing set, whereas the best value reported by the developer of the YOLOv5s model is 0.56 [41]. These results suggest that our pre-trained detection model performs moderately in detecting objects in images. This performance of the model is expected as we had to trade-off the system's accuracy for speed and computational time when choosing from different models.

4. Localization Algorithm for ADL Objects

Object localization is the identification of the position of one or more objects within an image or video frame. This involves detecting the object's presence and estimating its spatial coordinates, typically in a bounding box or mask [44]. Numerous computer vision and robotic systems rely on object localization to enable machines to perceive and interact with their surroundings. By localizing objects, robots can manipulate them with greater precision and efficiency, whether picking up and placing objects, maneuvering around obstacles, or grasping and manipulating objects. Accurate and reliable object localization is essential for robots to perform complex and sophisticated tasks in various environments.

Object localization plays a crucial role in developing assistive robots intended to aid individuals with disabilities or mobility impairments in completing daily tasks. When assisting users in everyday tasks, robots must accurately perceive and interact with objects in the environment, such as picking up objects or using tools. Object localization enables the robot to locate and manipulate these objects precisely, enhancing the efficiency and effectiveness of the robotic system, reducing the user's burden, and allowing them to perform tasks they might not be able to perform otherwise.

The novelty of our approach lies in developing a 3D localization algorithm for ADL objects within the context of assistive robots. Object localization is foundational for computer vision and robotic systems, allowing machines to effectively identify and interact with objects. It also fosters social interaction and engagement among individuals with disabilities. Robots equipped with object localization capabilities can actively participate in collaborative tasks with users, promoting user independence and fostering a sense of companionship.

4.1. Methodology

This section describes the methodology for developing our 3D localization system, which combines the ADL object detection model from Section 3 with localization, using a RealSense depth camera (D435) [45].

4.1.1. RealSense Depth Camera D435

To localize the detected objects in 3D space, we used an Intel RealSense Depth D435 camera [45]. The camera captures depth information in addition to color, which allows us to calculate the 3D position of objects in the scene. The Intel RealSense Depth (D435) camera works as a depth camera, using a combination of several sensors and algorithms [46]. The camera contains an RGB sensor, an infrared (IR) projector, an IR camera, and a dedicated depth processing chip (see Figure 2). When the camera operates, the IR projector emits a pattern of IR dots into the scene. The pattern of dots is designed to cover the entire scene and is structured in a way that allows the depth processing chip to calculate the depth of each pixel in the scene.

Figure 2. Intel RealSense depth camera D435.

The depth processing chip utilizes multiple algorithms to calculate the depth of each pixel [47]. One of these algorithms is "time-of-flight", which measures the time it takes for each IR dot to return to the camera after hitting an object in the scene. The depth processing chip uses these data to calculate the distance to each scene object. "Stereo vision" is another algorithm the chip uses for processing depth. This algorithm compares the IR dot pattern captured by the left and right IR cameras of the D435 camera to determine the image disparity. The contrast is then utilized to determine the depth of each pixel in the scene.

Once the depth processing chip has calculated the depth data, it is combined with the RGB data from the RGB sensor to produce a complete image containing depth information. This image has multiple applications, including 3D scanning, augmented reality, and robotics.

4.1.2. System Architecture

The 3D localization system described in this paper consists of three main components: an object detection model, a depth camera, and Intel RealSense SDK 2.0 [48] for object localization. The object detection model used in this system is a modified version of the YOLOv5s model which has been retrained on a dataset of ADL objects (see Section 3). The model was implemented using the PyTorch deep learning framework [49] and trained on a dataset containing objects commonly found in ADL scenarios, such as apples, oranges, cucumbers, and bottles.

This system employs a depth camera, which captures both RGB and depth data. RealSense SDK is used to access depth data and perform localization of objects. RealSense SDK provides a collection of APIs for accessing and processing depth data.

The localization algorithm follows a series of steps to complete the entire procedure. The system loads the pre-trained model for detecting ADL objects and initializes the depth camera by checking the connection between the camera and the system. If there is no issue with the connection, it starts reading RGB and depth frames from the camera. Then, the detection model takes the RGB frame as input and detects different ADL objects within the frame. The system then uses the bounding box information from the model to calculate the distance of the center point of the objects using the depth frame. The pseudocode of this localization system is illustrated in Algorithm 1.

Algorithm 1: The pseudocode of the localization algorithm

```
Initialization of detection model, display, and depth camera;
if The camera is connected then
    while If the exit command is not pressed do
        Take RGB and depth frames as input;
        Pass the RGB frame through the detection model;
        Assigns an index number to each object;
        Computes 2D coordinate of the target;
        Generate 3D coordinate using the depth frame;
        Displays the RGB frame with annotations and coordinates;
    end
end
```

The system then uses the x- and y-coordinates of the center point and the point's depth to generate 3D coordinates of different objects. Here, the x- and y-coordinates are in pixel value, and the z-coordinate (depth) is in millimeters. To use these coordinates for our application, we use a map function (in Section 5) to map all the coordinates into a single Cartesian coordinate system.

4.2. Results

After computing experiments with various objects, we found that the system could successfully detect and localize different objects, such as apples, oranges, capsicum, bottles, cups, etc., presented in front of the camera. After the detection model detects the ADL objects, the system generates the 3D coordinates in real time. Using this approach, the system can operate at 15–20 frames per second without issues. During the experiment, the depth frame accuracy is estimated as follows: 97.8% at short range (less than 50 cm), 93.4% at medium distance (51–100 cm), and 92.5% at larger distances (100–200 cm). A mean error of 4.8 cm is computed for the depth measurement throughout the experiment.

5. Design of the System Architecture to Perform the ADL Task

Performing ADL tasks is a crucial aspect of maintaining independence and improving quality of life. Assistive robots have the potential to improve the ADL performance of people with physical disabilities significantly. In this research, UFACTORY's xArm6 robot

is employed as an assistive robot manipulator [13]. This section presents the system architecture for semi-autonomously performing an ADL task using this assistive robot leveraging vision.

Our system utilizes the pre-trained object detection model from Section 3, which is trained to detect a wide range of ADL objects (as presented in Table 1). To localize these objects, we use the localization algorithm from Section 4 that integrates the output of the detection model with data from a RealSense depth camera. The RealSense depth camera provides detailed 3D information about the surroundings, allowing for precise object localization even in dynamic environments.

Once the system localizes any target object, the robot uses a distance-based control algorithm to manipulate the end-effector and approach the object. The system continuously monitors the distance between the end-effector and the target object, adjusting the robot's movement to minimize this distance. Once the end-effector is close enough to the object, the robot safely picks up the object, returns to its home position, and drops the object.

5.1. UFACTORY's xArm6

UFACTORY's xArm6 is a robotic manipulator designed for various applications, including as an assistive robot for individuals with physical disabilities. The xArm6 is a six-axis robotic manipulator programmed to perform various tasks, from simple pick-and-place operations to more complex manipulation tasks. This robot uses brushless motors with a 5 kg payload capacity and 0.1 mm repeatability. xArm6 is a high-precision, multifunction robotic manipulator with a 700 mm workspace.

One of the key features of the xArm6 is its high degree of flexibility and adaptability. The manipulator is designed to be easily integrated with various sensors and end-effectors, allowing it to be adapted to a wide range of applications. This adaptability makes it an ideal platform for assistive robots, as it can be adapted to meet the specific requirements of individual users.

The xArm6 is designed with safety in mind, in addition to its adaptability. The manipulator is equipped with various sensors and safety features to ensure that it can operate safely in close contact with humans. For instance, the robot is equipped with torque sensors that can stop the arm's movement if it collides with an object or a person to prevent injury. Overall, the xArm6 from UFACTORY is an example of an assistive robot that has the potential to improve the ADL performance of individuals with physical disabilities significantly. The arm's flexibility, adaptability, and safety features make it an ideal platform for our research, which aims to provide ADL assistance for performing "pick-and-place" tasks.

Kinematics and Dynamics of the xArm6 Robot

For the kinematic analysis, modified Denavit–Hartenberg (DH) parameters are used, whereas the iterative Newton–Euler approach is used for the dynamic analysis. Figure 3 shows the link-frame attachment for the modified DH parameters of the xArm6 robot, where the yellow circles indicate the direction heading into the viewing surface, the blue dot indicates the direction facing outwards from the surface, and the z-axis determines the rotation axis of each joint.

Table 2 illustrates the modified DH parameters for link-frame allocation to derive the forward kinematics of the robot [50]. Here, a, α, d, and θ, represent the common normal length, common normal angle, prior z-axis offset, and joint angle (in radians) of the robot.

Table 2. Modified DH parameters of the robot.

i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	267	0
2	0	$-\pi/2$	0	0
3	289.4886	0	0	0
4	77.5	$-\pi/2$	342.5	0
5	0	$\pi/2$	0	0
6	76	$-\pi/2$	97	0

Equation (2) presents the general form of the homogeneous transformation matrix (HTM) that correlates two sequential coordinate frames. Multiplying different transformation matrices generates the homogeneous matrix relating frame 6 to frame 0 (Equation (3)). Here, Equation (3) reflects the end-effector's frame locations and orientations relative to frame 0.

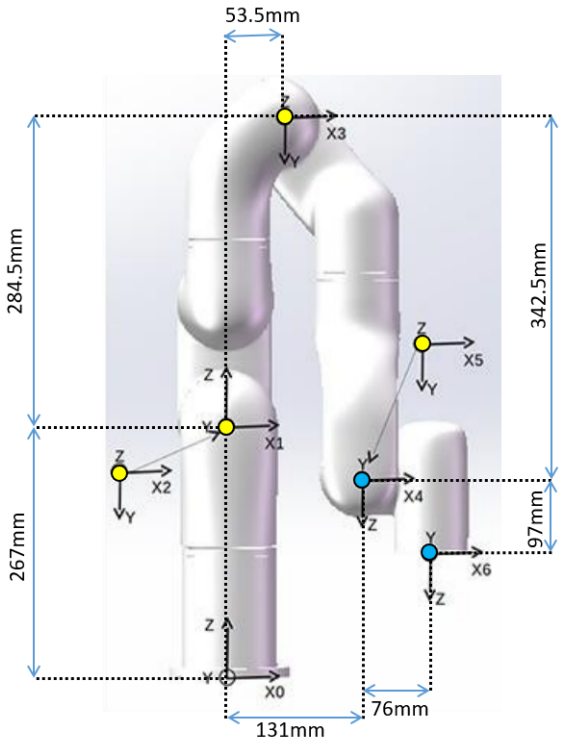

Figure 3. Modified DH parameters of the robot.

$$ {}^{i-1}_{i}\mathrm{T} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & \alpha_i \\ \cos\alpha_i \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i & -d\sin\alpha_i \\ \sin\alpha_i \sin\theta_i & \sin\alpha_i \cos\theta_i & \cos\alpha_i & d_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

$$ {}^{0}_{6}T = \left[{}^{0}_{1}T . {}^{1}_{2}T . {}^{2}_{3}T . {}^{3}_{4}T . {}^{4}_{5}T . {}^{5}_{6}T \right] \tag{3} $$

$$ \tau = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) \tag{4} $$

The xArm6 robot's dynamic formula is derived from the Newton–Euler equation, presented in Equation (4). Here, $M(\theta)$, $\ddot{\theta}$, $V(\theta, \dot{\theta})$, and $G(\theta)$ represent the 6×6 mass matrix of the manipulator, 6×1 acceleration vector, 6×1 vector of centrifugal and Coriolis terms, and 6×1 vector of gravity terms. Here, Table 3 illustrates the range of motion for each joint of the xArm6 robot.

Table 3. Range of motion for each joint of the xArm6 robot.

Joint *i*	Working Range
Joint 1	±360 (deg)
Joint 2	−118 to 120 (deg)
Joint 3	−225 to 11 (deg)
Joint 4	±360 (deg)
Joint 5	−97 to 180 (deg)
Joint 6	±360 (deg)
Maximum Speed	180 (deg/s)

5.2. Methodology

The proposed system executes a series of steps to complete the entire procedure. The system's pseudocode is illustrated in Algorithm 2. The system initially loads the YOLOv5s pre-trained model for detecting ADL objects. The model is loaded using PyTorch, which reads an image and returns an object list with bounding boxes. The system then initializes the display and depth camera and checks whether the camera is connected. If the camera is attached, the system initializes the robot object using the xArm-Python-SDK for further processing by using the robot's IP address to create a link between the robot and the system.

The system then begins processing the RGB and depth frames taken from the depth camera as input. First, the RGB frame is passed through the detection model. The model detects various ADL objects within the RGB frame and returns the bounding box information to the system. The system then displays the RGB frame with annotations to the user and checks whether or not the object pick-up mode is enabled.

If the object pick-up/tracking mode is disabled, the RGB frame, along with the object labels and bounding boxes for each object in the frame, is displayed to the user. Users can manipulate the robot's end-effector in any direction using the movement commands. Likewise, if the object pick-up/tracking mode is enabled, the system freezes the screen and annotates various objects before displaying the freeze frame to the user. It also stores the current location of the end-effector as the safe location to move back here after grabbing the target object. Additionally, the system assigns an index number to each object on the frame. Then, it asks the user which object should be picked up and takes the object's index number as input. Based on the user's input, the system then computes the 2D coordinate of the center point of the target object with respect to the depth camera. Utilizing the depth channel of the camera, the system estimates the distance between the camera and the center point of the target object and uses this as the third coordinate for 3D localization of the target object. Using the mapping equation (Equation (5)), it converts the coordinates from pixel values to robot Cartesian coordinates, where a, b, and c are constants and have different values for different axes.

$$x' = a * x^2 + b * x + c \tag{5}$$

As the distance estimation of the depth camera is not absolutely correct and can vary due to distance range or lighting, we do not use the 3D coordinate of the target object to manipulate the robot's end-effector directly. Instead, we use the 2D coordinate of the target object and manipulated the end-effector so that the target object's center point aligns with the RGB frame's center point. Then, we calculate the distance between the camera and the target object in real-time, estimate the distance error between them, and aim to minimize it by maneuvering the robot toward the object.

Algorithm 2: The pseudocode of the proposed system.

```
Initialization of detection model, depth camera, and xArm6 Robot;
if The camera is connected then
    while If the exit command is not pressed do
        Take RGB and depth frames as input;
        Pass the RGB frame through the detection model;
        Displays the RGB frame with annotations;
        if The object pick-up mode is enabled then
            Freezes the screen;
            Assigns an index number to each object;
            Input: Take index number of the target object;
            Computes 2D coordinate of the target;
            Generate 3D coordinate using the depth frame;
            Maps the coordinate to robot cartesian coordinate;
            Estimate the error between the end-effector and target;
            while The target object is not in the center point of the camera do
                Calculates the required adjustment to reduce error;
                Adjust the end-effector as needed;
            end
            while The distance between the target object and the camera is bigger than a
              threshold do
                Calculates the required adjustment to reduce distance;
                Adjust the end-effector as needed;
            end
            Move the end-effector to map camera-end-effector offset;
            Grab the target object;
            Move end-effector to the last command for safe dropping;
            Go to the home position of the robot;
            Drop the target object;
            The object pick-up mode ceases;
        else
            Input: Take movement command;
            Manipulate the end-effector according to the command;
        end
    end
end
```

All the manipulation of the end-effector is computed using the inverse kinematics function of the xArm6 manipulator provided by the xArm SDK. To change the end-effector's position, we read the current position values, adjust them, generate the desired position value for each step, and use the provided inverse kinematics function to reach the desired location. To ensure the robot's smooth adjustment, the end-effector's manipulation is restricted to one millimeter in each direction at a time. As of now, all the calculations are conducted with respect to the camera and the target object. To grab the object, the robot's end-effector needs to move based on the offset between the robot's end-effector and the camera. So, when the error exceeds a certain threshold, the system manipulates the robot based on the offset and grabs the object with a two-finger gripper.

To avoid any kind of collusion, the robot moves 5.0 cm upward after grabbing the target object and moves back to the safe location. Then, the robot moves to the home position of the system to drop the target object. After dropping the target object, the object pick-up mode ceases, and the robot is now free to be manipulated using any external command. Three different commands are also introduced among various controls: home, error, and emergency. The home command moves the robot back to its home position.

Whereas the error command clears any errors during the manipulation, and the emergency command terminates the whole session immediately and holds the robot's position.

From the user's perspective, if they wish to control the robotic arm manually, this can be achieved using a keypad or user interface; each key is dedicated to manipulating the robot in one direction. A basic user interface was designed (Figure 4) to perform the testing of the system using PySimpleGUI [51] library.

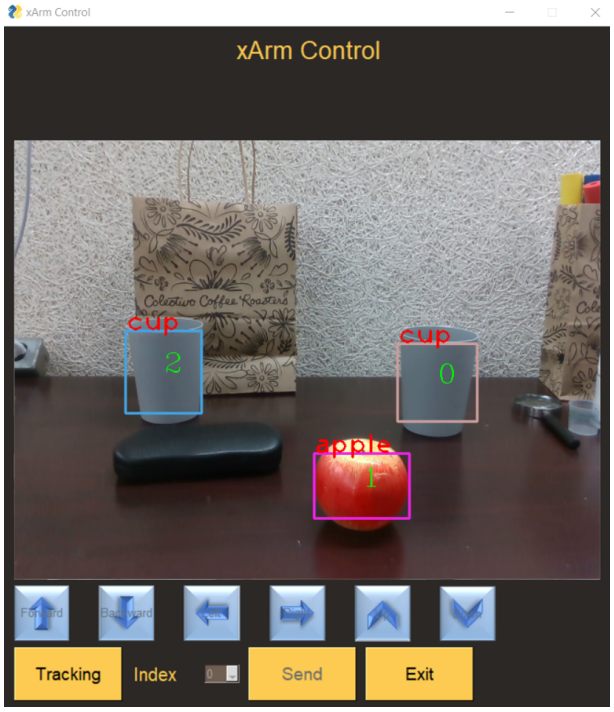

Figure 4. User interface of the system.

6. Experimental Implementation

This section outlines the experimental implementation of the proposed system architecture presented in Section 5, designed for a semi-autonomous assistive robot capable of performing a specific ADL task. The primary objective of this section is to demonstrate the effectiveness of the proposed system architecture in accomplishing the intended task.

6.1. Experimental Setup

In the conducted experiment, the vision sensor used for input is the Intel RealSense depth camera (D435), and the robotic manipulator employed is the xArm6. This USB-powered camera provides stereo depth information and precise depth sensing, featuring a global photo shutter and a wide field of view. UFACTORY offers a graphical user interface known as xArm Studio, along with software development kits (SDKs) for Python, Robot Operating System (ROS), and C++. In this research, we utilize the Python SDK to control the xArm6 robot. The robot's end-effector is equipped with a two-finger gripper, which can be replaced with other grippers such as a vacuum gripper, depending on specific application requirements. For safety, a physical emergency switch is incorporated into the robot.

Throughout both the development and experimental phases, the robot is affixed to a stationary table (as depicted in Figure 5). To simulate objects on an upper shelf during the experiments, various objects are positioned at four different locations, situated slightly away from the robot's base, as illustrated in Figure 5. These locations are characterized by different heights and horizontal offsets from the robot's base. Specifically, Location 1 has a height of 30.0 cm and is 63.5 cm from the robot's base, with a 7.5 cm offset to the right. Location 2 stands at a height of 43.0 cm and is positioned 53.5 cm from the robot's base,

with a 12.5 cm rightward offset. Similarly, locations 3 and 4 have heights of 38.0 cm and 33.0 cm and are situated 56.0 cm and 68.5 cm from the robot's base, with leftward offsets of 7.5 cm and rightward offsets of 18.0 cm, respectively.

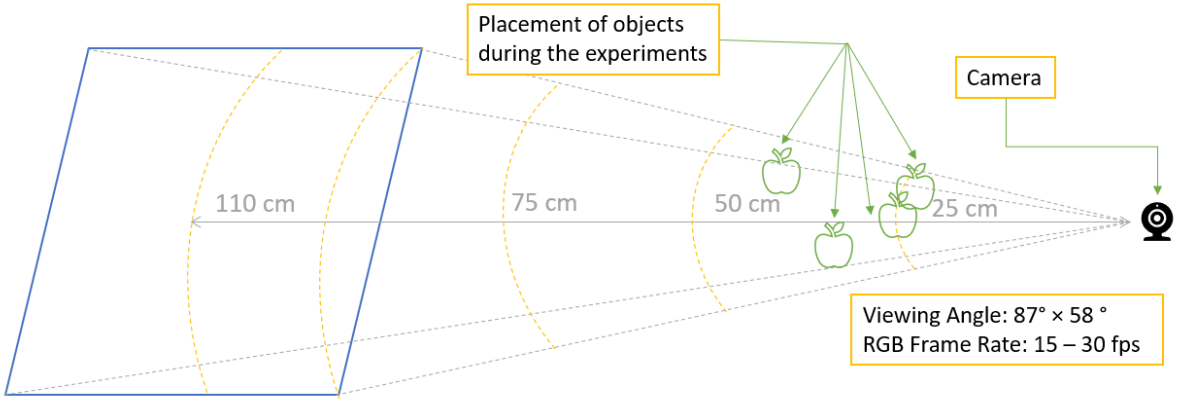

Figure 5. Experimental setup and locations of different objects during the experiment.

Figure 6 provides a schematic representation of the system setup. In this configuration, the robot connects wirelessly to a router via an Ethernet cable, facilitating communication between the robot and the control computer. The camera interfaces with the control computer via USB cables. A power source is linked to the robot's control box, which includes a physical emergency switch. The two-finger gripper on the robot's end-effector integrates a force sensor and is accompanied by the depth camera. All computational tasks are executed on the control computer.

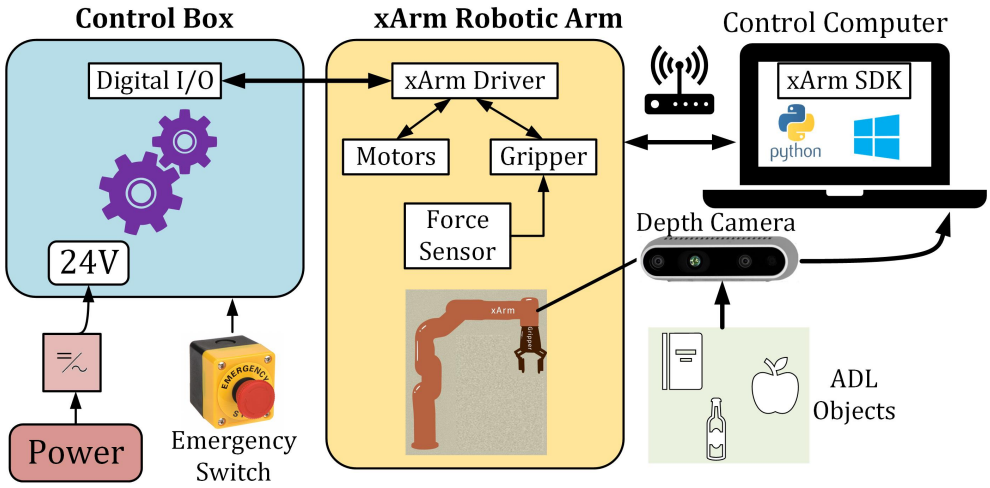

Figure 6. Schematic diagram of the proposed system.

6.2. Results and Discussions

The system's performance during experiments is displayed in Figure 7, which offers three viewpoints along with the position of the robot's end-effector. Each image sequence includes images captured by the depth camera, which is affixed to the robot's end-effector during the experiment, along with two additional viewpoints observing the system's performance. Here, the robot detects two ADL objects, an apple and a capsicum, presented in front of the robot, and displays labels and bounding boxes to the user. The user then enables the object pick-up/tracking mode and presses the target object's index number as input. Then, the system executes all the computations and tracks the target object. When the robot's end-effector approaches close to the target object and grabs that object, it moves

back to a safe location to avoid any collision and then goes to the home position to drop the object.

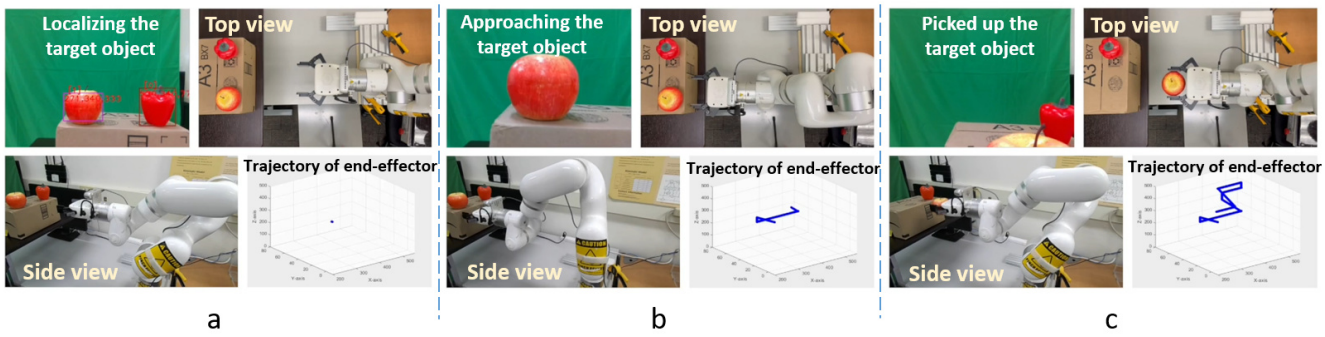

Figure 7. System performance from three different viewpoints, along with the end-effector's position during an experiment: (**a**) at the beginning of the experiment, (**b**) in the middle of the experiment, and (**c**) at the end of the experiment.

The experiments encompass diverse objects, including apples, oranges, cups, and capsicums, to assess the system's performance. The overall success rate of the system is computed to be 72.9%, with an average task completion time of less than 40 s after the thorough experiment. Completing this task with manual joystick control takes, on average, 62 s, with 27 input commands from the user. So, the proposed system demonstrates greater effectiveness and efficiency in task completion. We acknowledge that enhancements are necessary for broader real-world applications. The system's object recognition and localization aspects show promising performance; however, improvements are particularly needed in the grasping process and the precision of the depth camera. Future work will explore alternative camera technologies to enhance the system's reliability and applicability in diverse real-world settings.

The success rates of the system in different locations are displayed in Figure 8, whereas Figure 9 presents the success rate for each of the objects in different locations. From Figure 8, we can see that the overall success rate of the system in location 1 is 75%. Similarly, the success rates in locations 2–4 are 75%, 83%, and 58.33%. Figure 9 shows that the system works well for objects like apples or capsicums, whereas it performs poorly for objects like cups. The main issue behind this behavior of the system could be the object's shape. While the recognition and localization algorithm works well with different shapes of objects, the performance of the grasping strategy differs with the shape. For real-world applications, advancements are needed in the grasping process.

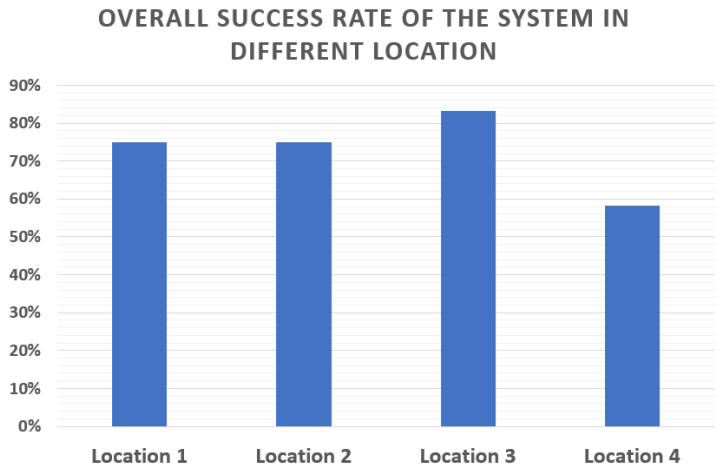

Figure 8. Success rate of the system in different locations.

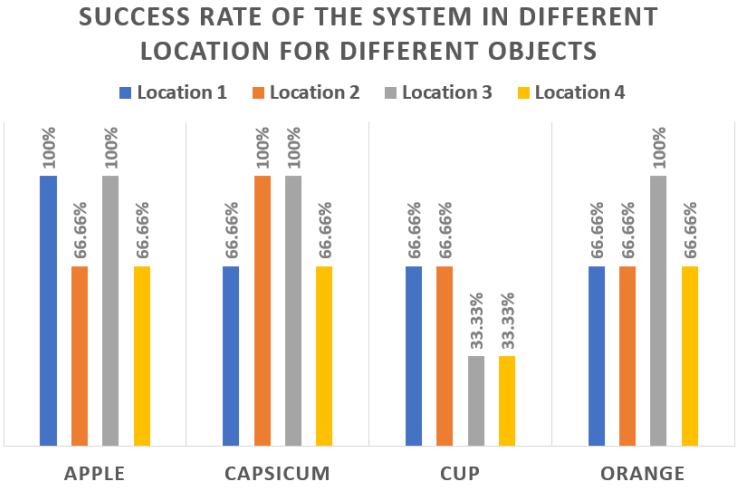

Figure 9. Success rate for each object in different locations.

From the experiments, we have also observed that, although the robot has a pretty large workspace, due to the position of the camera and the constraints of the algorithms, like moving upwards to avoid collisions or moving back to a safe location, the system can operate precisely within a window with a height of 25.0 cm to 50.0 cm and a distance of 50.0 cm to 75.0 cm from the base of the robot. In other words, the system can perform the pick-and-place ADL task semi-autonomously in this workspace, as shown in Figure 10. This range is suitable to help a wheelchair user to pick up ADL objects from an upper shelf.

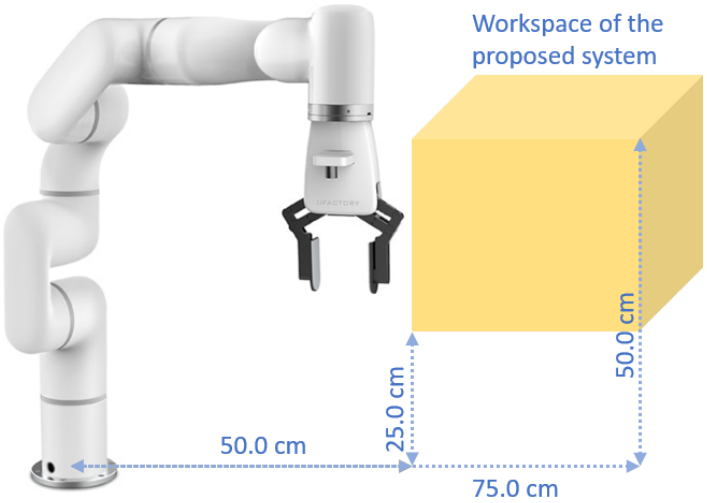

Figure 10. Operational workspace of the proposed system.

7. Conclusions and Scope for Future Work

This research aimed to design and implement a system architecture for an assistive robot that can perform daily living tasks for individuals with physical disabilities, enabling them to perform ADL tasks more independently and efficiently. The proposed system architecture utilized UFACTORY's xArm6 robot, a pre-trained object detection model based on YOLOv5s, and a RealSense depth camera to perform object localization and distance-based control. The methodology involved the retraining of the object detection model, the development of an object localization algorithm, the design of a distance-based control algorithm, and the integration of the system.

The experimental implementation demonstrated that the proposed system architecture has an overall accuracy of 72.9% in detecting, localizing, and performing one of the common ADL tasks, "pick-and-place" of ADL objects, making it suitable for ADL assistance. The proposed system architecture has the potential to positively impact the lives of individuals with physical disabilities by providing them with greater independence and autonomy in their daily lives. The system demonstrated high efficiency in performing the tasks, with an average task completion time of less than 40 s. This research has established a proof of concept for using a semi-autonomous system in robotic assistance for individuals with such needs.

In the future, we will continue our research and further develop the system's ability to perform additional ADL tasks autonomously. The insights gained from this study will be used to enhance the system's performance and practicality in real-world settings. The future work includes:

- Generate a labeled dataset of ADL objects from scratch using real objects to enhance the detection model's performance and explore alternative models.
- Explore alternative depth cameras and determine the most effective camera position that does not restrict the system's performance or workspace.
- Examine different approaches to control assistive robots and integrate orientation estimation into the model to enable semi-autonomous ADL task completion in complex environments.
- Integrate a user-friendly user interface with the system to offer more flexibility to the user.
- Incorporate the proposed control system into a wheelchair-mounted assistive robot and evaluate the performance with real users.

Author Contributions: Conceptualization, methodology, investigation, validation, formal analysis, data curation, visualization, writing—original draft preparation, M.T.S.; resources, supervision, project administration, M.H.R. and J.G.; writing-review and editing M.T.S., J.G., R.F. and M.H.R. All authors have read and agreed to the published version of the manuscript.

Funding: This material is based upon work partially supported by the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant number 90DPGE0018-01-00) and NASA-Wisconsin Space Grant Consortium (Award No. RIP23_1-0). NIDILRR is a Center within the Administration for Community Living (ACL), Department of Health and Human Services (HHS). The contents of this paper do not necessarily represent the policy of NIDILRR, ACL, or HHS, and you should not assume endorsement by the Federal Government or the National Aeronautics and Space Administration.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADLs	Activities of daily living
DoF	Degrees of freedom
ULEs	Upper and lower extremities
ULEDs	Upper and lower extremities dysfunctions
UE	Upper extremity
MVP	Minimum viable product

mR2A	Multifunctional Robotic Assistive Arm
3D	Three dimensional
CNNs	Convolutional neural networks
DCNN	Deep convolutional neural network
DNN	Deep neural network
GNN	Graph neural network
RNN	Recurrent neural network
SVM	Support vector machine
RL	Reinforcement learning
COCO	Common Objects in Context
mAP	Mean average precision
IoU	Intersection over union
DH	Denavit–Hartenberg
HTM	Homogeneous transformation matrix
SDK	Software development kit
ROS	Robot Operating System

References

1. Sitruk-Ware, R.; Bonsack, B.; Brinton, R.; Schumacher, M.; Kumar, N.; Lee, J.Y.; Castelli, V.; Corey, S.; Coats, A.; Sadanandan, N.; et al. Progress in progestin-based therapies for neurological disorders. *Neurosci. Biobehav. Rev.* **2021**, *122*, 38–65. [CrossRef] [PubMed]
2. Cai, Y.; Tian, Q.; Gross, A.L.; Wang, H.; E, J.Y.; Agrawal, Y.; Simonsick, E.M.; Ferrucci, L.; Schrack, J.A. Motor and physical function impairments as contributors to slow gait speed and mobility difficulty in middle-aged and older adults. *J. Gerontol. Ser.* **2022**, *77*, 1620–1628. [CrossRef] [PubMed]
3. Paralysis Statistics. 2023. Available online: https://www.christopherreeve.org/living-with-paralysis/stats-about-paralysis (accessed on 16 February 2023).
4. WHO EMRO|Stroke, Cerebrovascular Accident|Health Topics. 2023. Available online: https://www.emro.who.int/health-topics/stroke-cerebrovascular-accident/index.html (accessed on 16 February 2023).
5. Hussain, N.; Alt Murphy, M.; Sunnerhagen, K.S. Upper limb kinematics in stroke and healthy controls using target-to-target task in virtual reality. *Front. Neurol.* **2018**, *9*, 300. [CrossRef] [PubMed]
6. Carmeli, E.; Patish, H.; Coleman, R. The aging hand. *J. Gerontol. Ser. Biol. Sci. Med. Sci.* **2003**, *58*, M146–M152. [CrossRef] [PubMed]
7. Disability Impacts All of Us Infographic|CDC. 2023. Available online: https://www.cdc.gov/ncbddd/disabilityandhealth/infographic-disability-impacts-all.html (accessed on 16 February 2023).
8. The University of California-Disability Statistics Center. *Mobility Device Statistics: United States*; Disabled World: New York, NY, USA, 2024.
9. Goldman-Gerber, V.; Schwartz, I.; Rand, D. Upper extremity self-efficacy correlates with daily hand-use of individuals with high functional capacity post-stroke. *Disabil. Rehabil.* **2022**, *45*, 2301–2306. [CrossRef] [PubMed]
10. Kett, M.; Cole, E.; Turner, J. Disability, mobility and transport in low-and middle-income countries: A thematic review. *Sustainability* **2020**, *12*, 589. [CrossRef]
11. The Economic Impact of Caregiving. 2023. Available online: https://www.bcbs.com/the-health-of-america/reports/the-economic-impact-of-caregiving (accessed on 10 May 2023).
12. Jain, S.; Argall, B. Grasp detection for assistive robotic manipulation. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016 ; pp. 2015–2021.
13. Assistive Robotics for Activities of Daily Living|DO-IT. 2023. Available online: https://www.washington.edu/doit/programs/accessengineering/adept/adept-accessibility-briefs/assistive-robotics-activities-daily (accessed on 2 April 2023).
14. Moro, C.; Nejat, G.; Mihailidis, A. Learning and personalizing socially assistive robot behaviors to aid with activities of daily living. *ACM Trans. Hum.-Robot. Interact.* **2018**, *7*, 1–25. [CrossRef]
15. Lauretti, C.; Cordella, F.; Guglielmelli, E.; Zollo, L. Learning by demonstration for planning activities of daily living in rehabilitation and assistive robotics. *IEEE Robot. Autom. Lett.* **2017**, *2*, 1375–1382. [CrossRef]
16. Kim, D.J.; Lovelett, R.; Behal, A. An empirical study with simulated ADL tasks using a vision-guided assistive robot arm. In Proceedings of the 2009 IEEE International Conference on Rehabilitation Robotics, Kyoto, Japan, 23–26 June 2009; pp. 504–509.
17. Jivani, D. *Enabling Human-Machine Coexistence Using Depth Sensors*; Rensselaer Polytechnic Institute: Troy, NY, USA , 2020.
18. Try, P.; Schöllmann, S.; Wöhle, L.; Gebhard, M. Visual Sensor Fusion Based Autonomous Robotic System for Assistive Drinking. *Sensors* **2021**, *21*, 5419. [CrossRef]

19. Nguyen, T.H.C.; Nebel, J.C.; Florez-Revuelta, F. Recognition of activities of daily living with egocentric vision: A review. *Sensors* **2016**, *16*, 72. [CrossRef]
20. Villani, V.; Pini, F.; Leali, F.; Secchi, C. Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. *Mechatronics* **2018**, *55*, 248–266. [CrossRef]
21. Hellou, M.; Gasteiger, N.; Lim, J.Y.; Jang, M.; Ahn, H.S. Personalization and localization in human-robot interaction: A review of technical methods. *Robotics* **2021**, *10*, 120. [CrossRef]
22. Lanotte, F.; McKinney, Z.; Grazi, L.; Chen, B.; Crea, S.; Vitiello, N. Adaptive control method for dynamic synchronization of wearable robotic assistance to discrete movements: Validation for use case of lifting tasks. *IEEE Trans. Robot.* **2021**, *37*, 2193–2209. [CrossRef]
23. Zhong, J.; Ling, C.; Cangelosi, A.; Lotfi, A.; Liu, X. On the gap between domestic robotic applications and computational intelligence. *Electronics* **2021**, *10*, 793. [CrossRef]
24. Shahria, M.T.; Sunny, M.S.H.; Zarif, M.I.I.; Ghommam, J.; Ahamed, S.I.; Rahman, M.H. A Comprehensive Review of Vision-Based Robotic Applications: Current State, Components, Approaches, Barriers, and Potential Solutions. *Robotics* **2022**, *11*, 139. [CrossRef]
25. Yin, R.; Wu, H.; Li, M.; Cheng, Y.; Song, Y.; Handroos, H. RGB-D-Based Robotic Grasping in Fusion Application Environments. *Appl. Sci.* **2022**, *12*, 7573. [CrossRef]
26. Yu, Y.; Cao, Z.; Liang, S.; Geng, W.; Yu, J. A novel vision-based grasping method under occlusion for manipulating robotic system. *IEEE Sens. J.* **2020**, *20*, 10996–11006. [CrossRef]
27. James, S.; Davison, A.J. Q-attention: Enabling Efficient Learning for Vision-based Robotic Manipulation. *IEEE Robot. Autom. Lett.* **2022**, *7*, 1612–1619. [CrossRef]
28. Fang, W.; Chao, F.; Lin, C.M.; Zhou, D.; Yang, L.; Chang, X.; Shen, Q.; Shang, C. Visual-Guided Robotic Object Grasping Using Dual Neural Network Controllers. *IEEE Trans. Ind. Inform.* **2020**, *17*, 2282–2291. [CrossRef]
29. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
30. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
31. Kulkarni, N. Applications of Deep Learning based Object Detectors. 2020. Available online: https://www.einfochips.com/blog/applications-of-deep-learning-based-object-detectors (accessed on 27 April 2023).
32. Fakhry, A. The Applications and Benefits of a PreTrained Model—Kaggle's DogsVSCats. 2021. Available online: https://towardsdatascience.com/the-applications-and-benefits-of-a-pretrained-model-kaggles-dogsvscats-50221902c696 (accessed on 27 April 2023).
33. Brownlee, J. Impact of Dataset Size on Deep Learning Model Skill And Performance Estimates. 2020. Available online: https://machinelearningmastery.com/impact-of-dataset-size-on-deep-learning-model-skill-and-performance-estimates/ (accessed on 27 April 2023).
34. Open Data Essentials|Data. 2021. Available online: http://opendatatoolkit.worldbank.org/en/essentials.html (accessed on 27 April 2023).
35. ImageNet. 2023. Available online: https://www.image-net.org (accessed on 27 April 2023).
36. COCO-Common Objects in Context. 2021. Available online: https://cocodataset.org/ (accessed on 18 April 2023).
37. The PASCAL Visual Object Classes Homepage. 2020. Available online: http://host.robots.ox.ac.uk/pascal/VOC (accessed on 27 April 2023).
38. De Caro, J.D.S.; Sunny, M.S.H.; Montenegro, E.J.M.; Ahmed, H.U.; Rahman, M.H. Optimal Base Placement of a 6-DOFs Robot to Cover Essential Activities of Daily Living. *IEEE Access* **2022**, *10*, 134536–134548. [CrossRef]
39. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
40. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
41. Jocher, G. YOLOv5 by Ultralytics. 2020. Available online: https://github.com/ultralytics/yolov5 (accessed on 30 April 2023).
42. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv:2207.02696.
43. Solawetz, J. *What Is YOLOv5? A Guide for Beginners*. Roboflow Blog. 2023. Available online: https://blog.roboflow.com/yolov5-improvements-and-evaluation/ (accessed on 28 April 2023).
44. Borad, A. Understanding Object Localization with Deep Learning. 2021. Available online: https://www.einfochips.com/blog/understanding-object-localization-with-deep-learning (accessed on 28 April 2023).
45. Depth Camera D435. 2022. Available online: https://www.intelrealsense.com/depth-camera-d435 (accessed on 28 April 2023).
46. Tadic, V.L.; Sarcevic, P.; Sarosi, J.; Odry, Á.; Odry, P. RealSense Depth Sensors. 2022. Available online: https://encyclopedia.pub/entry/20328 (accessed on 30 April 2023).
47. RealSense Intel. Beginner's Guide to Depth (Updated). 2020. Available online: https://www.intelrealsense.com/beginners-guide-to-depth (accessed on 30 April 2023).
48. Pyrealsense2. 2023. Available online: https://pypi.org/project/pyrealsense2 (accessed on 28 April 2023).
49. PyTorch. 2023. Available online: https://pytorch.org (accessed on 1 May 2023).

50. Kinematic and Dynamic Parameters of UFACTORY xArm Series|UFACTORY Help Center. 2023. Available online: http://help.ufactory.cc/en/articles/4330809-kinematic-and-dynamic-parameters-of-ufactory-xarm-series (accessed on 25 December 2023).
51. PySimpleGUI. 2024. Available online: https://www.pysimplegui.com (accessed on 6 April 2024).