Department of Mechatronic Engineering

MXEN4000 - Mechatronic Engineering Research Project 1

Progress Report

Navigation Assistance for a Semi-Autonomous Smart Wheelchair

Semester 1, 2022

Jakob Wyatt
19477143

Supervisor(s): Yifei Ren & Siavash Khaksar

Abstract

Table of Contents

1 Introduction **1**

1.1 Aims 1

1.2 Problem Definition 2

2 Literature Review **3**

2.1 Sensors and Hardware 3

2.1.1 Sensor Types 3

2.1.2 RGB-D Cameras 3

2.1.3 Compute Element 4

2.2 Scene Understanding 6

2.2.1 Image Classification 6

2.2.2 Object Localization 6

2.2.3 Semantic Segmentation 7

2.3 Assistive Control 7

3 Methodology **8**

3.1 Hardware 8

3.2 Dataset Collection 9

3.3 Software 9

4 Current Work **10**

5 Future Work **11**

6 References **12**

List of Figures

1 CentroGlide in Reclined Configuration 1

List of Tables

1 Sensor Comparisons 3
2 Stereo Camera Options 4
3 AI Accelerator Options 5

1 Introduction

Many people with motor disabilities rely on wheelchairs for movement, and powered wheelchairs have enabled greater independance for people with disability. Despite the huge benefit powered wheelchairs have granted, the use of this technology can be inaccessible or unsafe for people with amyotrophic lateral sclerosis (ALS) or vision impairment, who may be unable to use a joystick or see their environment clearly.

1.1 Aims

The aim of this research is to develop a semi-autonomous smart wheelchair system. This research is done in collaboration with Glide, a WA wheelchair manufacturer, who have provided an existing powered wheelchair (CentroGlide) to use as a base for this functionality. By developing assistive technology for the wheelchair, the user is granted greater mobility, confidence, and independence.

Figure 1: CentroGlide in Reclined Configuration

1.2 Problem Definition

There are multiple engineering research project students who are part of this team, working on elements such as controller design, navigation assistance, and object detection. This work specifically focuses on pathway assistance, which identifies suitable paths for the wheelchair to drive on. If a user unintentionally drives off their desired path, this can lead to uneven terrain and possibly falling from the wheelchair. By guiding the user along a path, these safety issues can be mitigated.

Emphasis is placed on the 'semi-autonomous' aspect of the wheelchair. An important requirement of this project is that the user still has control over their wheelchair, and can override any semi-autonomous functionality if required. When false positives occur within the smart wheelchair system, the users mobility should not be compromised.

Another requirement of the system is that any sensors mounted to the wheelchair should not impede the users comfort or the wheelchairs manouverability. Many wheelchair users have specific requirements for wheelchair seat adjustments, to avoid pressure sores and discomfort. Figure 1 shows the wheelchair configuration when fully reclined, demonstrating that some sensor mounting locations are infeasible.

The smart wheelchair system should also be commercially viable - high-cost components and sensors are infeasible. Internet connectivity should not be a requirement for the system to operate either - the round trip time required to communicate with a server would compromise the safety of a user. Because of this, all processing is performed locally on the wheelchair.

2 Literature Review

2.1 Sensors and Hardware

2.1.1 Sensor Types

Smart wheelchairs have used a varied range of sensor types to percieve the surrounding environment. RGB-D stereo cameras have been widely used in the field [1][2][3], alongside 2d Lidar [4] and ultrasonic sensors [5]. Self-driving cars built by companies such as Tesla and Waymo use cameras, mmWave Radar, and 3d Lidar to avoid traffic and pedestrians.

Selecting a sensor to use is not necessarily an either-or decision. Sensor fusion algorithms such as the Extended Kalman Filter (EKF) or Unscented Kalman Filter (UKF) [6] allow outputs from multiple sensors to be used together to improve their accuracy. Additionally, sensors can be used for different applications on the smart wheelchair - a stereo camera could be used to sense the surrounding environment while an inertial measurement unit (IMU) could be used for wheelchair odometry. Table 1 gives a comparison of several available sensor types, taking into account factors such as resolution, cost, and accuracy.

Sensor	Advantages	Disadvantages
RGB-D Stereo Camera	Very high resolution	Low field of view (FOV)
mmWave Radar	High accuracy	Low resolution
3D Lidar	High resolution and accuracy	Very high cost
2D Lidar	High FOV and accuracy	Only detects obstacles within the same plane
Ultrasonic sensor	Low cost	One-dimensional

Table 1: Sensor Comparisons

2.1.2 RGB-D Cameras

One advantage RGB-D cameras have over alternative sensors is high RGB resolution, allowing them to utilize advances in machine learning and computer vision. Sensors such as LIDAR may fail at path detection, as features such as drawn lines must be distinguished visually from the surroundings.

When comparing these cameras, factors such as package size, field of view, and depth

accuracy are important to consider due to the available mounting points on the wheelchair. Several commercial options are compared in Table 2 - all of the listed units come with an integrated IMU.

Name	Type	Cost (AUD)[1]	Dimensions (mm)	FOV (Horizontal, Vertical, Depth)	Operating Range (m)
Stereolabs Zed Mini [7]	Passive	$595	$124.5 \times 30.5 \times 26.5$	$90° \times 60° \times 100°$	0.1-15
Stereolabs Zed 2 [8]	Passive	$670	$175 \times 30 \times 33$	$110° \times 70° \times 120°$	0.3-20
Intel RealSense D455 [9]	Active IR (Stereo)	$595	$124 \times 26 \times 29$	$90° \times 65° \times 87°$	0.6-6
Microsoft Azure Kinect DK [10]	Active IR (Time of Flight)[1]	$595	$103 \times 39 \times 126$	$75° \times 65° \times 75°$	0.5-3.86

Table 2: Stereo Camera Options

A caveat of the Stereolabs products is that they require a seperate CUDA enabled GPU (Nvidia) to generate the point-cloud and RGB-D image. The Kinect DK requires a CPU for processing, while the Intel RealSense performs processing onboard and only requires a USB-C 3.1 interface to communicate.

2.1.3 Compute Element

A compute element inside a semi-autonomous driving system generally consists of several components - a microcontroller to process user inputs and send signals to the motors, a general purpose computer to run pathfinding algorithms and log information, and an AI accelerator to improve performance of on-board ML algorithms.

AI accelerators use specialized hardware to perform operations common in ML algorithms (such as matrix multiplication and convolution) more efficiently than a CPU can. GPUs have been used widely for this application, however their high power usage is infeasible for some applications. Embedded AI accelerators aim to provide greater power efficiency at the cost of specialization.

ML models often use mixed-precision (FP16) datatypes to store weights while training. This helps improve the accuracy of the model, however can have a negative impact on speed during inference. One technique used to improve the speed of the model is quantization [11], where the FP16 datatype is replaced with an INT8 datatype (using a scaling factor and bias). This gives a large speed improvement, while only losing a small amount of model accuracy. For this reason, modern AI accelerators focus on the performance of

INT8 operations (TOPS, 10^9 operations per second), whereas earlier accelerators state the performance of FP16 operations (TFLOPS, 10^9 floating-point operations per second).

The Nvidia Jetson and Google Coral products are both popular options for embedded AI acceleration. These accelerators are compared in Table 3 alongside a gaming GPU.

Name	Cost (AUD)[1]	Release Year	Speed	Power	Notes
Nvidia Jetson Nano [12]	$150	Early 2019	0.5 TFLOPS (FP16)	10 W	-
Nvidia Jetson Xavier NX [13]	$595	Late 2019	21 TOPS (INT8)	20 W	-
Nvidia RTX 2080 [14]	$1040	2018	80.5 TFLOPS (FP16) 161.1 TOPS (INT8)	215 W	Doesn't include single-board computer
Google Coral Edge TPU [15]	$190	2020	4 TOPS (INT8)	2 W	Only supports TensorFlow Lite

Table 3: AI Accelerator Options

2.2 Scene Understanding

Scene understanding is a broad field, and involves using computer vision methods on visual or spatial data to gain better knowledge about the surrounding environment. Convolutional Neural Networks (CNNs) are commonly used for this application, as they are able to exploit the local nature of image features to reduce the number of required computations.

2.2.1 Image Classification

Image classification is a core problem within this field, and involves identifying the subject of an image (such as an animal or object). AlexNet [16], based on the earlier digit-recognition CNN LeNet-5 [17], was one of the first deep CNNs applied to this problem. Alexnet was trained on the large ImageNet dataset [18], which consists of 15M images and 22K categories, and achieved an error of only 15.3% on a 1000 class subset. The underlying architecture uses a series of 5 convolutional layers and 3 fully connected layers.

Neural network architectures have become deeper and more accurate over time, enabled by both growth in computational power and dataset size. VGG-16 [19] and GoogLeNet [20] are 16 and 22 layers deep respectively, and approached human performance on the ImageNet dataset. ResNet [21] is up to 156 layers deep, and exceeds human performance at image classification with an error of 3.57%. ResNet uses a 'skipping' architecture to improve network training, where the output of a layer relies directly on the input of a previous layer.

2.2.2 Object Localization

Object localization is another core problem within this field, and involves identifying the location of objects within an image, as well as classifying them. This is important for our wheelchair, as we want to be able to identify the location of a pedestrian or obstacle within the environment. R-CNN [22] was one of the first object classification models which utilized convolutional networks, by identifying potential bounding boxes and running an image classifier on these bounding boxes. Fast and Faster R-CNN [23][24] improved the speed of this model by running an image classifier backbone once on the entire image,

and using a CNN to improve identification of bounding boxes. Pascal VOC [25] and MS COCO [26] are datasets which are commonly used to evaluate object classification models.

YOLO (You Only Look Once) [27][28][29][30] is another object classification model which focuses on improving speed. In particular, YOLOv4 [30] reaches over 60 fps on the Tesla V100, which enables its use in real time applications such as autonomous driving and security camera footage. YOLO divides an image into an $S \times S$ grid, and uses a single convolutional network to output both bounding box predictions and image classification for each grid square. Low-probability and overlapping bounding boxes are then removed before the final output.

2.2.3 Semantic Segmentation

2.3 Assistive Control

3 Methodology

3.1 Hardware

The smart wheelchair should have the ability to sense, process, and manouver within the surrounding environment. To do this requires some necessary hardware, including a sensor system, compute element, and motor controller. Due to the 2021-2022 chip shortage, hardware selection was identified as a process that should occur relatively quickly.

The literature review provides a comparison between different sensor types and models. For outdoor navigation assistance, a forward facing RGB-D camera was selected as the best option for this project - specifically, the Zed 2 Mini. This was chosen due to the high resolution provided by RGB-D cameras, and the long distance range of the Zed cameras due to their passive operation (does not use an IR emitter).

The front of the joystick control unit was selected as the best mounting point for the stereo camera, due to several reasons:

1. Clear view of the environment in front of the wheelchair.
2. Not obstructed by the user in any wheelchair configuration.
3. When needed, the user can move the joystick control unit out of the way, which also moves the camera out of the way.

However, there are some challenges faced when using this mounting point, which must be addressed.

1. Shaky video footage due to low rigidity in joystick mount.
2. Close to the front of the wheelchair, which reduces visibility of the sides of the wheelchair.
3. Maximum camera width of 150 mm before doorway manouverability is affected.

Due to the size constraints, the Mini form factor was chosen.

An additional sensor with a wider field of view will be needed for doorway navigation and wheelchair docking, due to the limitations of this mounting point. The sensor selected

for this purpose was a 2D Lidar unit attached to the base of the wheelchair. However, this sensor will not be used for our specific application (pathway assistance).

For the compute element, a Nvidia Jetson Xavier NX will be used, due to compatability with the Zed API and a wide range of ML frameworks, as well as low power use. The motor controller is being designed by a different thesis student, however will connect with the compute element in a standard way.

3.2 Dataset Collection

To train and evaluate machine learning models, a 34 minute driving dataset was collected around Curtin University. The camera used to collect this dataset was a GoPro Hero 4. Although this camera does lack a depth component, having a basic dataset for model evaluation is valuable prior to procurement of the RGB-D camera.

3.3 Software

Our software system requires several components. Interaction with the Zed 2 Mini will be done via the proprietry Stereolabs API, which will provide RGB-D imaging to the computer. PyTorch [31] will be used for training and evaluation of machine learning models, as it provides good compatability with existing models such as YOLOv5 [32] and DeepLabv3 [33]. OpenCV [34] will be used to encode and decode video.

4 Current Work

5 Future Work

6 References

[1] H. Wang, Y. Sun, R. Fan, and M. Liu, "S2P2: Self-Supervised Goal-Directed Path Planning Using RGB-D Data for Robotic Wheelchairs," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11 422–11 428, 2021. DOI: 10.1109/ICRA48506.2021.9561314. [Online]. Available: https://sites.google.com/view/s2p2.

[2] H. Wang, Y. Sun, and M. Liu, "Self-Supervised Drivable Area and Road Anomaly Segmentation Using RGB-D Data For Robotic Wheelchairs," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4386–4393, 2019, ISSN: 2377-3766. DOI: doi.org/10.1109/LRA.2019.2932874.

[3] S. Jain and B. Argall, "Automated perception of safe docking locations with alignment information for assistive wheelchairs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2014, pp. 4997–5002. DOI: 10.1109/IROS.2014.6943272.

[4] M. Scudellari, "Self-driving wheelchairs debut in hospitals and airports [News]," *IEEE Spectrum*, vol. 54, no. 10, pp. 14–14, 2017, ISSN: 0018-9235. DOI: doi.org/10.1109/MSPEC.2017.8048827.

[5] S. Levine, D. Bell, L. Jaros, R. Simpson, Y. Koren, and J. Borenstein, "The NavChair Assistive Wheelchair Navigation System," *IEEE Transactions on Rehabilitation Engineering*, vol. 7, no. 4, pp. 443–451, 1999, ISSN: 1063-6528. DOI: 10.1109/86.808948.

[6] E. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, IEEE, 2000, pp. 153–158. DOI: 10.1109/ASSPCC.2000.882463.

[7] Stereolabs, *ZED Mini Camera and SDK Overview*, 2018. [Online]. Available: https://cdn.stereolabs.com/assets/datasheets/zed-mini-camera-datasheet.pdf.

[8] Stereolabs, *ZED 2 Camera and SDK Overview*, 2019. [Online]. Available: https://cdn.stereolabs.com/assets/datasheets/zed2-camera-datasheet.pdf.

[9] Intel, *Intel RealSense Product Family D400 Series Datasheet*, 2022. [Online]. Available: https://www.intelrealsense.com/wp-content/uploads/2022/03/Intel-RealSense-D400-Series-Datasheet-March-2022.pdf.

[10] Microsoft. "Azure Kinect DK Hardware Specifications." (2021), [Online]. Available: https://docs.microsoft.com/en-us/azure/Kinect-dk/hardware-specification.

[11] B. Jacob, S. Kligys, B. Chen, *et al.*, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," 2017. DOI: 10.48550/ARXIV.1712.05877. [Online]. Available: https://arxiv.org/abs/1712.05877.

[12] Nvidia, *Jetson Nano System-on-Module Data Sheet*, 2019. [Online]. Available: https://developer.nvidia.com/embedded/downloads.

[13] Nvidia, *Jetson Xavier NX Series System-on-Module Data Sheet*, 2019. [Online]. Available: https://developer.nvidia.com/embedded/downloads.

[14] Nvidia, *Turing GPU Architecture*, 2018. [Online]. Available: https://images.nvidia.com/aem-dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf.

[15] Google Coral, *Coral Dev Board Datasheet*, 2020. [Online]. Available: https://coral.ai/static/files/Coral-Dev-Board-datasheet.pdf.

[16] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, pp. 1097–1105, 2012, ISSN: 0001-0782. DOI: 10.1145/3065386.

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, ISSN: 0018-9219. DOI: 10.1109/5.726791.

[18] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014.

[20] C. Szegedy, W. Liu, Y. Jia, *et al.*, "Going Deeper with Convolutions," 2014.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2016, IEEE, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[22] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2013.

[23] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision*, vol. 2015, IEEE, 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," 2015.

[25] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2009, ISSN: 0920-5691. DOI: 10.1007/s11263-009-0275-4.

[26] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft COCO: Common Objects in Context," 2014. DOI: doi.org/10.48550/arXiv.1405.0312.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2015. DOI: doi.org/10.48550/arXiv.1506.02640.

[28] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2016. DOI: doi.org/10.48550/arXiv.1612.08242.

[29] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018. DOI: doi.org/10.48550/arXiv.1804.02767.

[30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020. DOI: doi.org/10.48550/arXiv.2004.10934.

[31] A. Paszke, S. Gross, F. Massa, *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems*, vol. 32, pp. 8024–8035, 2019.

[32] Ultralytics, *YOLOv5*. [Online]. Available: https://github.com/ultralytics/yolov5.

[33] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," 2017. DOI: 10.48550/ARXIV.1706.05587. [Online]. Available: https://arxiv.org/abs/1706.05587.

[34] G. Bradski, *The OpenCV Library*, 2000. [Online]. Available: https://opencv.org/.