A Multimodal AI Approach for Intuitively Instructable Autonomous Systems: A Case Study of an Autonomous Off-Highway Vehicle

Abdellatif Bey Temsamani[1], Anil Kumar Chavali,
Ward Vervoort
Flanders Make
Lommel, Belgium
[1]abdellatif.bey-temsamani@flandersmake.be

Tinne Tuytelaars[2], Gorjan Radevski
KU Leuven, ESAT
Leuven, Belgium
[2]tinne.tuytelaars@esat.kuleuven.be

Hugo Van hamme[3]
KU Leuven, PSI
Leuven, Belgium
[3]hugo.vanhamme@esat.kuleuven.be

Kevin Mets[4], Matthias Hutsebaut-Buysse, Tom De Schepper, Steven Latré
University Of Antwerp - imec
Antwerpen, Belgium
[4]kevin.mets@uantwerpen.be

***Abstract*— In current production shop floors, a fleet of production machines and AGVs form a full manufacturing system with a high degree of automation. These current manufacturing systems need to deal with high variability of products and production tasks. Every task, however, requires a proper reconfiguration & control that is often done manually requiring complex settings and long configuration time. With AI techniques the reconfiguration of these systems to deal with a new task can be made more intuitive. In this paper we present the upgrade of an autonomous system, used for manufacturing assets handling and transportation, with AI features that make it easy to reconfigure in order to deal with high variability of assets and missions. Visual and spoken information is used to instruct and guide the autonomous vehicle using an AI multimodal framework where first, spoken language, with different local dialects, is translated to digital instructions, that can be associated to visual information to form control instructions to the autonomous vehicle. Different AI models, respectively for spoken language understanding, visual perception, vision based navigation are associated through a multimodal AI framework to intuitively control the AGV to perform a specific task. Beside the challenges related to the integration of these models in the AGV platform, other challenges related to dealing with variabilities of dialects, objects, surroundings and ambient conditions are partly tackled in this research.**

Keywords-AI based autonomous systems; Multimodal AI; Natural language processing (NLP); deep learning; neural networks; reinforcement learning

I. INTRODUCTION

Autonomous Guided Vehicles (AGVs) are becoming more and more popular in industrial applications. They can pick up and deliver materials around a manufacturing facility or warehouse [1]. However, with the continuously increase of mass customization [2], a return on investment of production AGVs can only be obtained if these AGVs can easily perform large variability of tasks and / or deal with large variability of products.

Tasks scheduling and allocations have been done by a central entity for a fleet of AGVs following predefined configurations. Driven by flexibility, robustness and scalability requirements, the current trends in AGV systems are customization and decentralization [3]. In a decentralized architecture, an AGV broadcasts the information about its states in a local way and decides which actions to take [4]. Although new generation of AGVs are highly instrumented with different guidance systems (optical, magnetic, laser, etc.), they are more optimized and suited for long-distance transportation of materials from / to multiple destinations, and / or tuned for repetitive and predictable tasks [5].

(Re-)configurating AGVs to perform multiple tasks in a non-predictable environments remains, however, a challenge today in industrial floors due to dynamically changing environments. Literature in this research remains very limited and focuses more on path planning methods in unknown environments, yet using references (e.g. markers, identifiers, etc.) [26]. Research on voice controlled AGV remains in the level of performing basic operations (e.g. moving with constant speed) in a prescribed path [27].

In this paper, we propose a Multimodal Artificial Intelligence (AI) framework that allows to intuitively and easily (re-)configure an AGV to perform different and variable tasks. In this framework, an operator can instruct the AGV by speech interaction that can be done locally or remotely. The operator can intuitively instruct the vehicle. This instruction is then decoded through different interpretation layers that make respectively use of **(i)** natural language processing, **(ii)** association with vision deep learning for objects recognition and localization and **(iii)** association with reinforcement learning for navigation.

(i) Spoken interaction offers fast and natural interaction with machines and AGVs, while operators keep their hands and eyes free for other tasks. The task of a Spoken Language Understanding (SLU) component is to map speech onto an interpretation of the meaning of a command, while taking the variability in the input signal into account: differences in voice, dialect, language, acoustic environment (noise, reverberation), hesitation, filled pauses and pure linguistic variation. Traditionally, SLU is approached as a cascade of Automatic Speech Recognition (ASR) mapping speech into text followed by Natural Language Understanding (NLU) mapping text onto meaning. This cascaded approach tends to propagate and inflate ASR errors and requires application-specific textual data, which is unnatural to acquire. Instead, this work uses End-to-End SLU (E2E SLU), where spoken instructions are directly mapped onto meaning without textual intermediate representations.

 ISBN: 978-1-61208-966-9

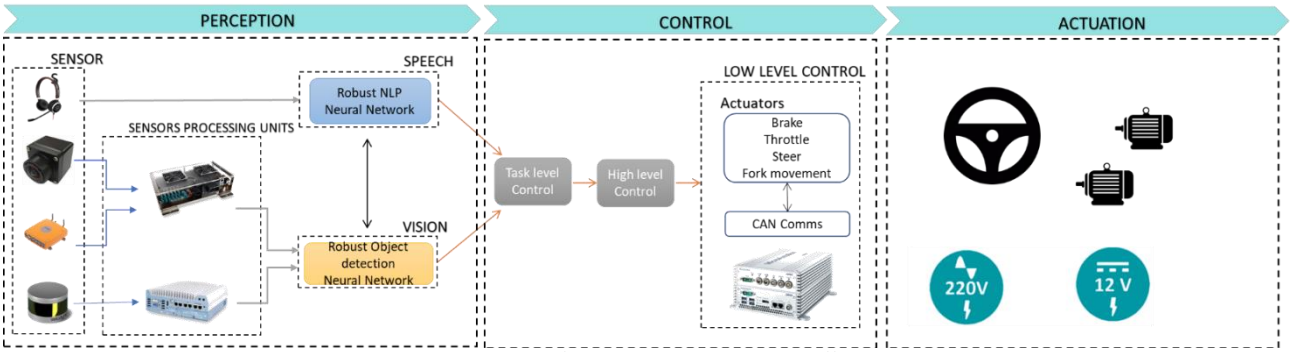

Figure 1. Autonomous platform around the autonomous off-highway vehicle

(ii) For the agents to interact with the environment, they must process and understand visual input, i.e., extract the semantically relevant cues from the environment in order to execute the desired task. Should the input be provided from an RGB camera, a plethora of Deep Learning techniques could be leveraged to achieve visual understanding. Deep Learning techniques rely on Neural Networks, commonly (pre-)trained on large-scale general-purpose datasets, e.g., for visual recognition [6] such as object detection [7]. Since our goal is to interpret a language-based instruction we need to locate the object(s) referred to by the person in the environment. To this end, we build on a state-of-the-art object detection method. Given an RGB input, the object detector's role is to locate (detect) the relevant objects, i.e., potential objects that the person instructing the AGV might refer to. This serves as a backbone to perform multimodal interaction — associating the representation of the language-based instruction with the representation of the spatial layout of the scene (2D location and categories of the detected objects).

(iii) Egocentric navigation is one of the core problems intelligent systems need to master. An agent needs this skill not only to execute the task at hand, but also to navigate, in order to collect experience that can be used to learn from. Navigation is typically done by either using expensive specialized lidar and radar sensors, or by relying on visual inputs. In this paper we examine the performance of utilizing an RGB camera, a depth camera or a combination of both for navigation. In the presented approach we have chosen for an end-to-end learning-based navigation approach. Such an approach is able to outperform Simultaneous Localization and Mapping (SLAM) based approaches [8], it doesn't suffer from propagation errors due to mapping errors and excels in visually sparse environments [9]. As we need to train our navigation system in simulation due to the large amount of required interactions with the environment, we also propose a digital-twin based solution to utilize a navigation model trained in simulation in the real world. Through our multimodal speech and vision system, combined with learned navigation, we demonstrate an intuitively instructible autonomous system, which can act as a platform for various tasks.

II. CASE STUDY – AUTONOMOUS OFF-HIGHWAY VEHICLE

The architecture of the autonomous vehicle, its HW / SW components and upgrades with the Multimodal AI framework are described in this section.

A. Autonomous vehicle architecture

The AGV used in this paper consists of the off-highway tractor developed at Flanders Make [10]. The architecture of the AGV consists of a perception, control, and actuation frameworks (Figure 1). To perceive the environment we use cameras, lidars, a GNSS system and a microphone. The sensors data is then processed in separate computing platforms and stored on middleware (ROS), from where the Speech and Vision units send the information to the control block. This later is divided in two levels, (i) a High-level controller that controls the tractor via a state machine and (ii) a Low-level controller, built in a dSpace platform [11], that controls the trajectory such that velocity and heading can be followed. The output signals are sent to different actuators that consist of the brakes, throttle, steering and fork implement that are controlled via servo motors. Autonomous vehicle upgrades to deal with Multimodal AI

An example of intuitive instructions given by an operator to the AGV to execute a task and their high level interpretations by the Multimodal AI framework, described in this paper, is illustrated in Figure 2.

The instruction: '*Pick up the red pallet and put it on the truck*', needs first to be communicated to the computer that runs the speech AI module (described in Section III). In the next level, a vision module, where real time 2d vision data is processed and fed to a pretrained NN, allows objects classification and their association to different attributes such as object's type, color, etc. (as described in Section IV). The AGV should then move towards the recognized object. This step is supported by the association made so far between speech and vision data as well as the navigation data. This later makes use of the cartesian coordinates of the AGV in the navigation space and the reinforcement learning module (as described in Section V) that allows to estimate the optimal trajectory between the AGV and the object of interest.

In order to implement and demonstrate the Multimodal AI framework, The AGV is updated by a newly installed system for interfacing through speech with a dedicated PC. This PC is also used for developing and testing the neural

 ISBN: 978-1-61208-966-9

networks. It is equipped with a powerful Nvidia GPU and a new headset microphone for giving audio commands. The autonomous tractor internally uses ROS to communicate between the different sub-systems. Originally it was only used sparingly in the autonomous tractor, mainly to communicate lidar sensor data. After the system upgrade, also the control unit, the dedicated PC and the Nvidia Drive platform have a ROS interface. While the Nvidia Drive could technically runs the neural networks, for more convenience, during testing we installed the neural networks on the dedicated PC. Data from the cameras on the Nvidia Drive, LiDAR and navigation all come in as ROS messages while for speech a simple microphone is connected to the PC. The output of the multi-modal setup is the location of a specific object together with the task the tractor must complete. This information can be communicated through ROS to the navigation module.

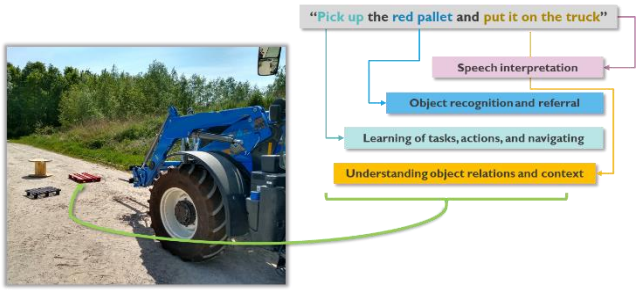

Figure 2. Example of speech-based instruction and multimodal mapping

III. SPOKEN LANGUAGE UNDERSTANDING

This section summarizes the speech data generation, speech model training and testing as well as the architecture and the validation of the spoken language understanding.

A. *Speech data generation*

To train the SLU model, training dataset with audio fragments is made. It is important that the recorded speech seems natural, as if the participants are really interacting with the AGV. To this end, we believe that a visual feedback to the participant would be very useful. Therefore, a simple automotive simulator called Webots [12] was used and a set of API calls were written in order to control the simulated tractor in the simulated environment (Figure 3).

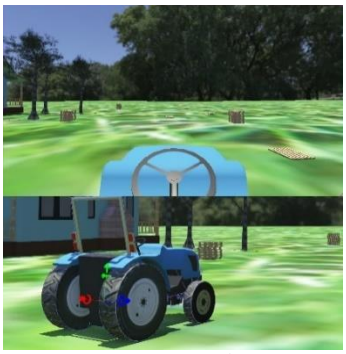

Figure 3. Simulator that provides visual feedbacks to participants for speech recording

The participants are given some high-level objectives and it is up to them to control the tractor with speech commands in order to fulfil these tasks. With the '*high-level*' objectives (in contrast with explicitly providing the primitive commands to the participants) we aim to improve the variability of commands that participant's would naturally choose to control the tractor. Every time the participants speak a relevant command, the experiment supervisor presses a button to invoke the correct API call. This way, we already have some automatically generated annotations linking the participant's speech command to the supervisor's API call invocation. We recorded the audio in Audacity in WAV format using a headset microphone and a separate standalone microphone. The commands were mainly basic control commands like turning a direction or driving speed. A total of 14 people who speak *Dutch* language (different dialects) were recorded with mixed female and male voices.

B. *SLU model architecture & training*

Classical *semantic frames* are used for representing the semantics of an utterance. A semantic frame is composed of *slots* (e.g. "direction") that take one of multiple *slot values* (e.g. "forward" or "backward"). This encoding represents the affordances of the AGV and corresponds to API calls with parameters filled in. The task of the SLU component is to map an utterance (spoken command) to a completed semantic frame. The SLU architecture follows the encoder-decoder structure first described in [13] and later refined in [14] to allow for encoder pretraining for ASR targets on generic *Dutch* data. The decoder is trained on the task-specific data. The encoder encodes an utterance in a single high-dimensional embedding in two steps. The first step maps MEL-filterbank speech representations to letter probabilities using a transformer network [15] preceded by a down sampling CNN, trained maximal cross-entropy between predicted and ground truth transcriptions in a 37-letter vocabulary. The training data consist of 200 hours of Flemish speech with its textual transcription from the CGN corpus [16], fourfold augmented with noise (0-15 dB) and reverberation (sampled from [28]) to achieve acoustic robustness. The second step counts bigram occurrence frequencies of all letter pairs across the utterance and repeats the same while skipping one position in the bigram, resulting in a $2(37^2) = 2738$ dimensional utterance embedding.

The decoder maps the utterance embedding onto a multi-hot encoding of the slot values via non-negative matrix factorization (NMF) [17] as described in [13]. Other than in the pretraining stage, the training pairs here do not require textual transcription, but are pairs of speech with the completed semantic frame. Here, a neural network could be taken as well, but the chosen decoder has several advantages: (1) it requires few training data, (2) it retrains in a fraction of a second when user interaction data becomes available and (3) it establishes a bag-of-words model making the SLU system less sensitive to the rather free word order in *Dutch* (at least compared to *English*). Learning a stricter word order would require more task-specific training data exhibiting the word order variability.

The approach is evaluated on the Grabo corpus [18], which contains a total of 6000 commands to a robot spoken by ten *Flemish* speakers and one *English* speaker. The

commands were recorded with the participants' own hardware in a quiet room at their homes. The semantics are described in eight different semantic frames describing driving, turning, grabbing, pointing, … using one (e.g. "*close* gripper") to three (e.g. "*quickly* drive *forward a little bit*") of ten slots (e.g. angle, direction, …), which can take between two and four different values. In total, 33 different meanings occur in the data. The accuracy is evaluated as the F1-score for slot values as a function of the number of task-specific training examples. The trained decoder is speaker-specific. The average accuracy over speakers is plotted in Figure 4 and shows that with the minimal of 33 training utterances, i.e. one example per meaning, an accuracy of over 98.5% is reached. The performance saturates around 180 task-specific utterances.

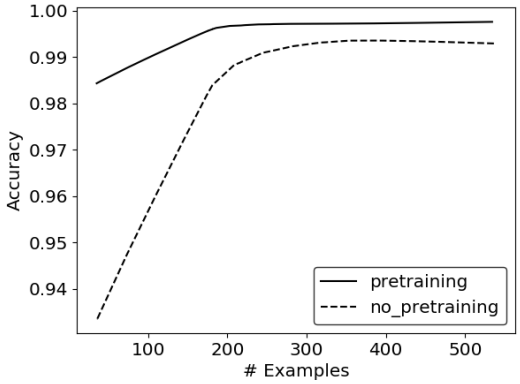

Figure 4. F1-score as a function of the number of task-specific training examples.

C. SLU model validation

For deployment we set up a docker container to run all the code. We developed a user interface to be able to easily visualize the results of the SLU model and provide training examples for training the decoder. In this interface, it is possible to record samples, open the microphone so the tractor can listen, give feedback to the model and retrain the model. After each command is given the confidence value of the prediction is estimated. Commands with sufficient confidence are forwarded to the tractor through ROS to the control PC.

The initial accuracy of the model depends a lot on the person giving the commands and their accent. But we were able to achieve high levels of accuracy of more than 90 percent in the noisy tractor environment using an active learning approach. In this approach, the operator can give feedback samples to retrain the model. In this experimental set-up, repeating an instruction in 5 instances proved to achieve high accuracy (90%). The retraining flow is quite time-efficient and takes less than a second to retrain.

IV. VISUAL PERCEPTION AND ASSOCIATION WITH SPEECH

The visual perception of the scene, including the scene data generation, the AI model architecture, it's training and testing as well as its association with speech in the Multimodal AI framework are described in this section.

A. Vision AI Objects detection and classification

1) Vision data generation

The dataset for training the vision model contains images with mostly objects that the AGV can pick up. This means mostly pallets and boxes of varied materials, shape and sizes containing materials like bobbins and wooden planks. This data was recorded on the Flanders Make local site, spread over two occasions: once on an early cloudy morning in spring and one just after noon in summer with sunny weather. Every image was recorded with a resolution of 960 x 608 pixels. The entire dataset contained 1100 images, derived from 9 videos. These videos each recorded one configuration of objects from many angles.

2) Vision NN architecture & training

The main building block of the vision pipeline is the object detector. It gets an RGB image I as input, where $I \in \mathbb{R}^{3 \cdot H \cdot W}$ and H and W are the image height and width respectively. The model we use is a state-of-the-art two-stage object detector, where in the first stage, a region proposal network generates regions of interest for the image, and in the second stage, bounding boxes and object classes are predicted for each proposal, which exhibits an objectness score above a certain threshold. The region proposal network generates region proposals by sliding a spatial window over a feature map obtained from a Convolutional Neural Network (CNN), i.e., a backbone. Additionally, the object detector includes a Feature Pyramid Network [19], a fully-convolutional module, which generates feature maps at different levels, thus enabling the model to recognize objects at different scales. The object detector we use is a Faster R-CNN [20], with a ResNet101 backbone [21], pre-trained for general purpose object detection on COCO [7]. Even though less resource intensive Faster R-CNN backbones exist, such as MobileNets [29], given our computational budget, we find the Faster R-CNN variant we use to yield the best tradeoff between detection performance and speed (near real-time).

The model's outputs are object bounding boxes and classes with a confidence score for each. The confidence score for the predicted class is obtained as the Softmax probability of the highest scoring class.

We perform fine-tuning of the Faster R-CNN on images consisting of scenes from the environment, where the objects of interest are annotated with bounding boxes and classes. The images we use are video frames, extracted from 9 videos of the AGV navigating the environment while encountering the objects. Considering that the amount of data at our disposal is limited, we have to ensure that the model does not overfit on some, irrelevant properties of the data, e.g., the weather, the relation between objects' position in the frame and their categories, etc. To decrease the influence of these components, and to attempt simulating a diverse evaluation environment to a certain extent, we determine the optimal hyperparameters by training the object detector in a leave-one-out fashion. Namely, we train on a subset of 8 videos and perform evaluation on the remaining one. We iterate this process until we train a separate model on all unique subsets. The final model performance is averaged over each of the videos. We evaluate the model's performance using the

 ISBN: 978-1-61208-966-9

standard COCO [7] mean average precision (mAP). The final model, i.e., the model used in the AGV, is trained on all 9 videos using the hyperparameters determined during the leave-one-out training/evaluation process.

We train the model for 5 epochs with a learning rate of 1e-4. We perform random horizontal flip data augmentation, enabling us to synthetically increase the dataset size and make the detector invariant to such data transformations. We sample a subset of 128 region proposals to estimate the regression and classification loss of the region proposal network.

We quantitatively evaluate each of the trained models on the videos, which were held-out during training (Table 1). The lowest score is highlighted in red, while the highest scoring one in green. Overall, taking the current state-of-the-art of COCO as a reference point (~60% mAP when writing this paper), we observe that the performance is relatively high across all different videos (57.3 mAP). We further observe that the performance on Video 2 (Vid. 2), is significantly lower compared to the average performance. To inspect the reason for the lower performance, we qualitatively inspect the samples from Video 2 as discussed below.

TABLE 1. QUANTITATIVE EVALUATION OF THE VISION AI TRAINED MODEL

	Vid. 1	Vid. 2	Vid. 3	Vid. 4	Vid. 5	Vid. 6	Vid. 7	Vid. 8	Vid. 9	Avg.
mAP	55.04	40.90	56.03	66.42	68.35	50.50	65.25	61.9	51.42	57.30

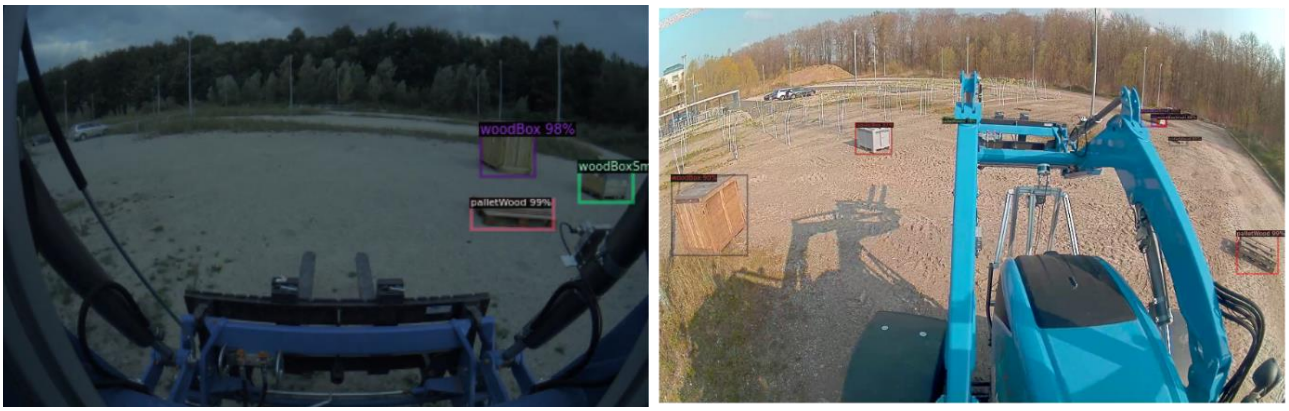

Figure 5. (left) all objects are correctly classified, (right) some objects are not detected

We qualitatively evaluate the object detector's performance by visualizing the predictions on the held-out videos during training. In Figure 5 (left), we observe that the model correctly predicts all objects, which is in line with our expectations as the objects are fully visible and of a reasonable size. On the other hand, in Figure 5 (right) we observe several mis-detected objects of a frame from Video 2. We conclude that even though the model performs well, it struggles to recognize objects, which are (1) far from the camera (small size), and (2) occluded in the environment – both of which are active areas of object detection research.

B. Visual grounded SLU

To deal with the data sparsity, and to be able to ground (localize) the speech model output in the image, we perform discretization of the spatial layout (the bounding boxes and classes obtained as output from the object detector). To be specific, we perform mapping of both modalities to a canonical space, where we later measure the similarity between the output of the speech model and each of the detected objects in the image. To that end, we encode each detected object as a collection of one-out-of-k encodings of its category (box, pallet, etc.), material (wooden, plastic, etc.), size (regular or small), and location in the image. Note that the object category, material and size are jointly predicted by the object detector as the object class. Namely, since the number of possible category + material + size combinations is limited to 6 in our use-case, we encode each combination as a separate class. Lastly, we want to emphasize that such approach does not scale well as the number of possible object categories, materials and sizes increases, however, we leave the decoupling as future work.

Lastly, we quantize the location of the object, i.e., we represent the object's location based on the object's horizontal and the lower vertical position. We showcase the grid over the image including the spatial references according to the x and y axis in Figure 6.

Finally, we represent each detected object as a vector of size 12, where we allocate 3, 2, 3, 3, 1 indices for the object's class, material, x-location, y-location and size respectively. When measuring the similarity between the speech model output (a vector of size 12 as well) and each encoded object detection, we explore different weighting strategies for each object attributes which we discuss next.

C. Adding Spatial relations

We evaluate different strategies for measuring the similarity between each (discretized) object detection and the

 ISBN: 978-1-61208-966-9

speech model output. The output is a bounding box, which represents the grounding location of the instruction. We evaluate each grounding strategy on two variants of the dataset, namely (1) a descriptive variant, where the objects are commonly described based on their attributes, e.g., pick up the wooden box, and (2) a spatial variant, where the referred object is described based on its location in the frame, e.g., pick up the box furthest on the left.

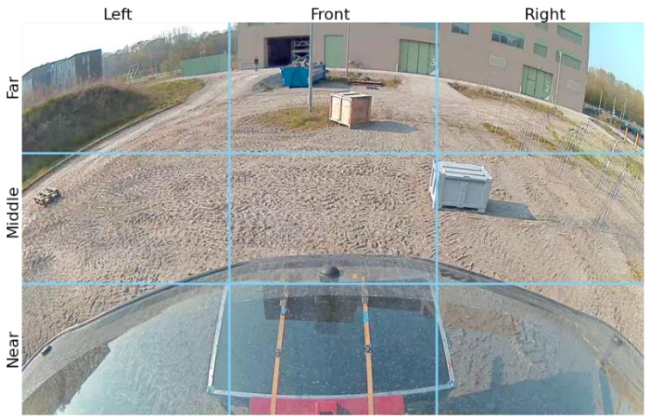

Figure 6. Grid over image with object's spatial reference

The grounding strategies we evaluate are:

1. Random matching (RM): A naïve baseline, where we ground the speech given instruction to a randomly selected bounding box. We establish a lower bound on the grounding performance with this baseline.
2. Basic matching (BM): We obtain the dot-product between the one-hot encodings of speech instruction and each object detection, representing the similarity.
3. Weighted matching (WM): We (re-)scale the contributions of the individual elements in the dot-product with pre-defined weights.
4. Confidence matching (CM): We represent the speech model with the confidence scores.
5. Weighted confidence matching (WCM): We use confidence scores for the speech model output and additionally weight the individual contributions using the pre-determined weights.

We perform evaluation using the standard grounding accuracy metric, where we score a hit if the predicted grounding bounding box has intersection over union (IoU)>0.5 with the ground truth box. For the random baseline, we perform inference 5 times and report the average performance. For the grounding strategies which weight the importance of each attribute (WM, WCM), we perform grid search over various weight combinations, and establish a weight of 0.1, 0.7, 0.2, and 0.05 for the spatial indicators, the object class, the object material and the object size respectively. We hypothesize that the success of this particular weight combination is a result of (1) the object class and material are essential to ground/locate the referred object, (2) the spatial indicators are somewhat imprecise, but still indicative of the location of the object, and (3) the object size mostly depends on the distance between the AGV camera and the object, which makes it noisy, and should be down weighted. We report the results in Table 2.

TABLE 2. EVALUATION OF THE AI MODEL WITH SPATIAL RELATIONS

	Dataset type	
Method	Descriptive	Spatial
RM	25.91	17.14
BM	65.91	59.52
WM	70.45	57.94
CM	76.14	62.70
WCM	**79.55**	**65.87**

We observe consistent gains when we weigh (WM) or use the speech model confidence scores (CM) in the grounding, compared to the baseline basic matching (BM) method. Additionally, a combination of the weight and confidence matching (WCM) yields superior results across the different data (descriptive, spatial) and significantly outperforms the other methods. Lastly, even though the spatial data is more challenging than the descriptive data, the WCM module performs well, indicating that by re-weighting and adding confidence scores, we can ground spatial speech data reasonably well.

V. REINFORCEMENT LEARNING BASED NAVIGATION & ASSOCIATION WITH SPEECH-VISION DATA

In this section, the navigation part of the Multimodal AI and it's association with the speech-vision data is described. The currently developed proof of concept consists of a simulation environment with the hardware in the loop.

To make this simulator as close to real life as possible, a 3-D scan of the test environment by using an aerial scanning using a drone with photogrammetry capabilities that allows us to map images to a high fidelity 3-D twin of the area. This twin was then imported to the simulator for the purpose of reinforcement learning.

 ISBN: 978-1-61208-966-9

1) RL architecture & training

The presented Reinforcement Learning (RL) approach makes use of the DD-PPO (Decentralized Distributed Proximal Policy Optimization) architecture [22] (Figure 7).

The Reinforcement Learning (RL) approach is able to map high dimensional inputs to discrete actions. The DD-PPO model consists of a visual pipeline, for which in our case we use a ResNet18 [21].

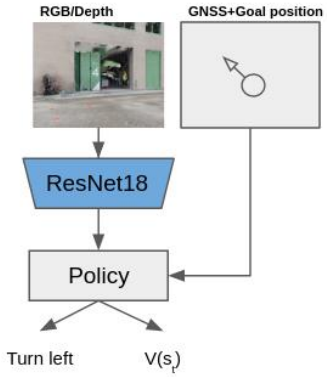

Figure 7. DD-PPO architecture overview

The resulting learned visual representation is concatenated together with a GNSS sensor. This output is then passed onto a recurring policy consisting of 2 Long short-term memory (LSTM) [23] layers. The final outputs of the model consists of a state value estimation, and an action distribution from which actions (move forward, turn left, turn right and stop) can be sampled. The stop-action should be executed by the agent when positioned less than 2 meters of the goal position. As inputs for the model we tested a single depth camera, a single RGB camera, or a combination of both RGB and depth. We use these sensors as they are cheap and widely available. The camera is positioned on the front of the AGV.

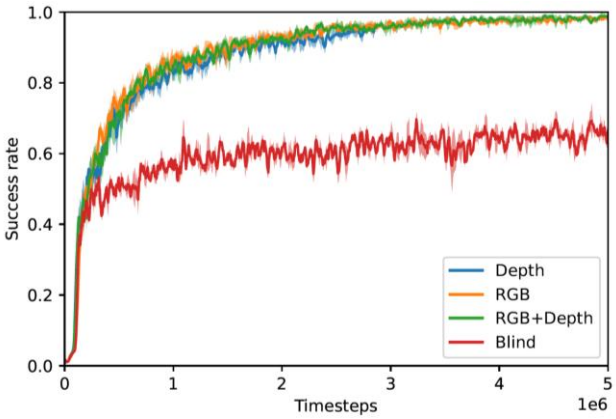

Figure 8. Training performance. The blind agent can perform basic navigation by relying on the GNSS sensor, however to further improve to near perfect results an additional RGB of depth sensor is required to detect and avoid collisions.

To train the agent we use the improvement in geodesic distance between the agent and the goal position as a dense reward signal. A slack penalty of -0.01 is subtracted on each step, and a termination bonus of 2.5 is awarded upon successfully utilizing the done action. We train the agent entirely in the Habitat simulator [8] where a photorealistic scan of the environment is used. This allows the agent to interact with the terrain in a safe way. While in this case we trained the agent to specifically work on a single environment, DD-PPO also allows generalization to unseen environments, given enough different training environments and training samples. Figure 8 shows the required number of interactions with the environment. These results indicate that in this setting the agent relies mostly on the GNSS sensor, as the blind agent performs reasonably (60% success rate after 5M training interactions). However, by adding either a depth or RGB sensor the agent achieves near perfect navigation capabilities on the training set after 5M interactions with the simulated environment.

2) RL validation

Realizing Reinforcement learning on a large autonomous platform brings in multiple challenges to the board. For safety concerns, the approach to validate the system was to use a Hardware-in-loop setup along with the digital twin of the environment. The main input from the real world was the signal from the GNSS receiver (Septentrio AsteRx-U) on the AGV, which was then mapped to the digital twin coordinates system. The GNSS had a dual antenna setup which could then provide the heading of the platform as well. Using a cloud-based service updates were provided in real time to the simulator/digital twin environment to position the simulated tractor same as the one in real world. The output from the simulator was the suggested trajectory to the goal pose.

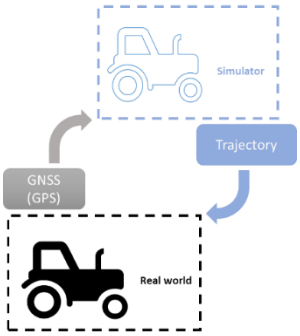

Figure 9. Hardware In Loop setup (overview).

To evaluate the navigation capabilities of the agent, we created a holdout dataset. This holdout dataset contains goal positions the agent did not see during training. Table 3 contains the results of 100 tested episodes. In Table 3, the success rate indicates the amount of episodes the agent could complete successfully. The Success weighted by Path Length (SPL) measurement also considers the length of the path taken.

TABLE 3. SUMMARY OF TESTED EPISODES

Sensors	**Success Rate**	**SPL**	**Avg. Collisions**
RGB	100%	0.9454	0.4355
Depth	100%	0.8882	0.1129

 ISBN: 978-1-61208-966-9

RGBD	100%	0.9272	0.5161
Blind	91.94%	0.7294	4.3548

VI. AGV Multimodal AI Demonstration

To demonstrate the full methodology, we combined the methods respectively described in Sections III, IV and V in one demonstrator implemented in the AGV. We added all the information in a new docker environment to be able to run on the dedicated PC in the AGV. There is a similar user interface compared to the ***Speech*** (NLP) model (Section III) where you can record your voice and use the NLP model to predict the voice commands. These commands consist of the description of the object and the task the AGV should do. Then the fusion model uses this information to link an object description with a detection from the ***Vision*** model to predict the location of the describer object on the image. As a last step the lidar data is used to link the 2D location on the image to a 3D location of the object in the world coordinate space. This location can then be sent further as a goal to the control systems together with the described task from the NLP model. A significant improvement could be made in the parameters of the fusion model. There was a bias against using spatial information in the voice command. The material of the object is more difficult to extract on the image than its location, so using the location for finding the correct object is more reliable. Hence, we tuned some of the weights to have a bigger focus on this kind of information. Another small improvement could be made to the audio side. The person dedicated to controlling the AGV added some voice samples and gave feedback to the model through the user interface. This way the model was more confident in recognizing their accent and way of talking. With regards ***Navigation***, although the approach is not fully implemented in the rea system, the approach can already be demonstrated by Hardware-In-the-Loop. In this setting an instance of the simulator is constantly synchronized with the AGV. This is done by using the GNSS position from the real-world AGV to set the position of the agent in the simulator. We can use the digital twin to generate trajectory paths. These generated trajectories can then be used in the real-world by the AGV. A snapshot from the full demonstrator is depicted in Figure 10.

VII. Conclusion

In this paper, we developed and demonstrated a Multimodal AI framework that allows to intuitively instruct production AGVs to perform multiple tasks. The interface with operators is allowed by speech interaction that is decoded through an AI NLP model to translate to interpretable instructions both by AGV controller and the other components of the AI Framework. Associations with Vision and Navigation data is done respectively through an AI detector ad classifier model that recognize different types of objects in a varying environment, as well as a Reinforcement Learning model that estimate the optimal trajectory between the AGV and the objects. The Multimodal AI framework proves to work in different varying conditions where the AGV (Autonomous tractor) is configured to perform different outdoor missions (handling different types of objects such as boxes, pallets, etc.) under different ambient conditions (sunny, rainy, day, night, etc.). The demonstrator remains however a research proof of concept (to demonstrate the approach) and requires different improvements before an effective industrial usage. This includes amongst others, training with larger datasets (speech, vision, navigation) and evaluation in extended number of scenarios. Our research will continue on demonstrating this Multimodal AI approach in other industrial applications where AGVs are typically used, such as in Logistics.

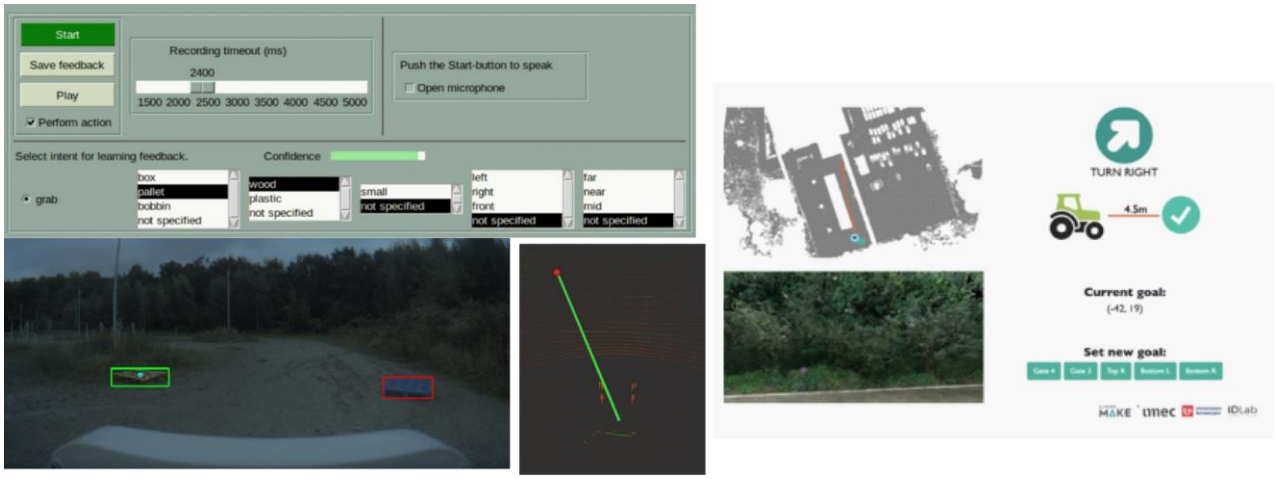

Figure 10. Snapshot of the demonstrator of the AGV Multimodal AI framework: (*top left*), the ***Speech*** model interface, (*bottom left*), the ***Vision*** model interface, (*right*), the ***Navigation*** digital twin interface, (*bottom middle*), the estimated trajectory between the AGV and the object that is dynamically calculated by fusing all models together

 ISBN: 978-1-61208-966-9

ACKNOWLEDGMENT

This research is supported by AI Flanders that is financed by EWI (Economie Wetenschap & Innovatie) (https://www.flandersairesearch.be/en) where all the authors of this publication are collaborating to perform AI research for different applications including industrial applications, and Flanders Make (https://www.flandersmake.be/en), the strategic research Centre for the Manufacturing Industry who owns the AGV infrastructure. The authors would like to thank everybody who contributed with any inputs that support to make this publication.

REFERENCES

1. Li, Dong, Bojun Ouyang, Duanpo Wu and Yaonan Wang. “Artificial intelligence empowered multi-AGVs in manufacturing systems.” *ArXiv* abs/1909.03373 (2019): n. pag.
2. Radder, Laetitia & Louw, Lynette. (1999). Mass customization and mass production. The TQM Magazine. 11. 35-40. 10.1108/09544789910246615.
3. M. De Ryck, M. Versteyhe, F. Debrouwere, Automated guided vehicle systems, state-of-the-art control algorithms and techniques, Journal of Manufacturing Systems, Volume 54, 2020, Pages 152-173.
4. Herrero-Perez, D. & Barberá, Humberto. (2008). Decentralized coordination of automated guided vehicles. 1195-1198.
5. Mousavi M, Yap HJ, Musa SN, Tahriri F, Md Dawal SZ (2017) Multi-objective AGV scheduling in an FMS using a hybrid of genetic algorithm and particle swarm optimization. PLoS ONE 12(3): e0169817.
6. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. DOI:https://doi.org/10.1145/3065386.
7. Lin *et al*., “Microsoft COCO: Commeon objects in Context’’, European Conference on Computer Vision, ECCV 2014, p. 740-755
8. Lin, TY. *et al.* (2014). Microsoft COCO: Common Objects in Context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham. https://doi.org/10.1007/978-3-319-10602-1_48
9. Savva, Manolis & Kadian, Abhishek & Maksymets, Oleksandr & Zhao, Yili & Wijmans, Erik & Jain, Bhavana & Straub, Julian & Liu, Jia & Koltun, Vladlen & Malik, Jitendra & Parikh, Devi & Batra, Dhruv. (2019). Habitat: A Platform for Embodied AI Research.
10. Mishkin, Dmytro & Dosovitskiy, Alexey & Koltun, Vladlen. (2019). Benchmarking Classic and Learned Navigation in Complex 3D Environments.
11. Automated off-highway vehicle test platform. [Online]. Available form : https://www.flandersmake.be/en/testing-validation/product-validation/automated-off-highway-vehicle-test-platform
12. Real-time testing system (dSpace). [Online] Available from: https://www.dspace.com/en/pub/home.cfm
13. Open source robot simulator. [Online]. Available from: https://cyberbotics.com/
14. Bart Ons, Jort F. Gemmeke, Hugo Van hamme, Fast vocabulary acquisition in an NMF-based self-learning vocal user interface, Computer Speech & Language, Volume 28, Issue 4, 2014, Pages 997-1017,
15. Wang, Pu & Van hamme, Hugo. (2021). Pre-training for low resource speech-to-intent applications.
    arXiv:2103.16674
16. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.
17. Oostdijk, Nelleke. (2000). The Spoken Dutch Corpus: Overview and first evaluation. Proceedings of LREC-2000, Athens. 2.
18. Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In Proceedings of the 13th International Conference on Neural Information Processing Systems. MIT Press, Cambridge, MA, USA, 535–541.
19. ALADIN: Adaptation and Learning for Assistive Domestic Vocal Interfaces. [Online]. Available from:
    https://www.esat.kuleuven,be/psi/spraak/downloads/
20. T. -Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 936-944, doi: 10.1109/CVPR.2017.106.
21. Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans Pattern Anal Mach Intell. 2017 Jun;39(6):1137-1149. doi: 10.1109/TPAMI.2016.2577031. Epub 2016 Jun 6. PMID: 27295650.
22. K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.
23. Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, Dhruv Batra, (2019, September). DD-PPO: Learning Near-Perfect PointGoal Navigators from 2.5 Billion Frames. In *International Conference on Learning Representations*. arXiv:1911.00357
24. Sepp Hochreiter, Jürgen Schmidhuber; Long Short-Term Memory. *Neural Comput* 1997; 9 (8): 1735–1780. doi: https://doi.org/10.1162/neco.1997.9.8.1735
25. Kang Tong, Yiquan Wu, Fei Zhou, Recent advances in small object detection based on deep learning: A review, Image and Vision Computing, Volume 97, 2020, 103910, ISSN 0262-8856.
26. M. Majdi, M. Deldar, R. Barzamini and J. Jouzdani, "AGV Path Planning in Unknown Environment Using Fuzzy Inference Systems," *2006 1ST IEEE International Conference on E-Learning in Industrial Electronics*, 2006, pp. 64-67, doi: 10.1109/ICELIE.2006.347213.
27. H T, Sreenivas & C, Arjun. (2017). Design of Voice Controlled Automated Guided Vehicle.
28. Aachen Impulse Response Database. [Online]. Available from: https://www.iks.rwth-aachen.de/en/research/tools-downloads/databases/aachen-impulse-response-database/
29. Howard, Andrew G., Zhu, Menglong, Chen, Bo, Kalenichenko, Dmitry, Wang, Weijun, Weyand, Tobias, Andreetto, Marco and Adam, Hartwig MobileNets: Efficient Convolutional Neural Networks for Mobile Vision. Applications. (2017). , cite arxiv:1704.04861

 ISBN: 978-1-61208-966-9