2014 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
AUTONOMOUS GROUND SYSTEMS (AGS) TECHNICAL SESSION
AUGUST 12-14, 2014 - NOVI, MICHIGAN

Implementing Robotic Control Algorithms in Open Source and Government Virtual Environments

Philip Frederick
US ARMY TARDEC
Warren, MI

Dr. Mike Del Rose, PhD
US ARMY TARDEC
Warren, MI

David Pirozzo
US AMRY TARDEC, Warren, MI
Dr. Robert Kania, PHD
US ARMY TARDEC, Warren, MI
Bernard Theisen
US ARMY TARDEC Warren, MI
Stephanie Roth
US ARMY TARDEC, Warren, MI

ABSTRACT

There are an increasing number of simulation tools either available today or coming online for ground vehicle autonomous behavior modeling. Engineers in TARDECs Ground Vehicle Robotics (GVR) area have dedicated a significant portion of effort over the past four plus years reviewing, implementing and promoting the development these tools. This paper provides an overview of these efforts and discusses the current state of these virtual environments from the perspective of TARDEC robotic researchers, developers, engineers and program managers. We specifically touch on experiences implementing control algorithms in Player/Stage, ANVEL and MODSIM/RIDE. While we take specific measures to avoid comparative analysis we provide our insight into the current strengths and limitations of these systems as it relates to utilization on research efforts, a quick development/integration program (Dismounted Soldier Autonomy Tools) and a JCTD effort (AMAS). We also give an overview of other known autonomy M&S systems and their stated status/attributes as well as discuss what we believe would be necessary for these tools to mature to the point where they can eventually serve to reduce costs associated with autonomous vehicle development and testing.

INTRODUCTION

Unmanned Ground Vehicle (UGV) autonomy continues to be an area of significant interest to the US Department of Defense. However, after the investments made in the Army's FCS (Future Combat Systems) ANS (Autonomous Navigation System) program and recent US conflicts abroad the current and near term funding allocations do not align with this interest. It is within this domain that research labs and organizations must identify methods to succeed on advanced autonomy maturity and development efforts with reduced funding. An area frequently seen as low hanging fruit is with the concept of a virtual proving ground for autonomous systems. It is envisioned that such a capability could reduce costs, and time of development, of these systems while also increasing the maturity and overall acceptance of the autonomous unmanned ground vehicle end product.

This paper intends to give an overview of the current state of the Autonomous Virtual Proving Ground (AVPG) concept from the perspective of engineers, researchers, program mangers and developers within TARDECs Ground Vehicle Robotics (GVR) area. First we will give some background on the AVPG concept over the years. Next we will give an overview of the facets of what we believe are important in an AVPG. After that we will give an implementation history of AVPGs in GVR including our experiences working with and around these systems. Finally we will finish with a brief conclusion statement about the state of these systems.

AVPG Background

Modeling and simulation has proven to be a key component of the development cycle for any complex system or process. The development of control and decision algorithms for Unmanned Ground Vehicles (UGVs) clearly falls into the category of complex systems. One of the first endeavors into robotic control algorithm simulation came via free ware as part of the Player project [1]. This simulation project quickly became popular with academia and small businesses as a means to test conceptual singular and multi-agent behaviors in virtual environments without the need to entirely recreate unique and specific interface modules for sensors, vehicles, and environments. Other simulation efforts began at this time as well, such as Webots [2], that went on to eventually form the basis for a few online programming competitions (such as the robostadium).

In the mid 2000’s the ARMY Research Lab invested in the creation of its own robotic autonomy simulation environment with the creation of Robotic Interactive Visualization and Experimentation Toolbox (RIVET) [3]. This tool was built by ARL to enable their Collaborative Technology Alliance members to have access to virtual robotic systems to enhance their autonomy development and integration efforts. Also around this time the ARMY’s Future Combat Systems Autonomous Navigation Program created a robotic behavior simulation environment, MODSIM, which enabled them to streamline autonomy development and testing procedures as well as meet many of their Configuration Item Development Specification requirements virtually. Both of these systems were developed by the same company, General Dynamics Robotic Systems (GDRS), and for the same general purpose, robotic autonomy development and virtual testing. The difference was in intended use as RIVET was developed to further research efforts while MODSIM was developed to help with development, testing and requirement achievement for a program of record.

More recently there have been investments in these virtual testing environments by the Office of the Secretary of Defense (OSD) and the Corps of Engineers Engineering Research and Development Center (ERDC) with the Autonomous Navigation Virtual Environment Laboratory (ANVEL) [4] and the Virtual Autonomous Navigation Environment (VANE) [5]. ANVEL was originally designed to be the playback entity for the VANE off-line high fidelity simulation system. However, ANVEL has been transformed into its own game engine based, open source, real-time simulation environment. VANE is a high fidelity, near real-time physics-based, multi-scale numerical testbed designed to quantitatively predict sensor and autonomous system performance in a simulation environment. These systems are designed to work in coordination such that when the need for really high fidelity environmental or sensors representations are required, for specific areas in a large simulation scene, arrangements can be made to configure the systems to provide for that deep VANE based analysis.

Also in recent years there has been investment by NASA/JPL and DARPA to also develop these virtual robotic testing environments but with a broader focuses than just the autonomy execution of developed behavioral and control based algorithms. The JPL system, Rover Analysis, Modeling and Simulation (ROAMS) [6], was developed to support planet surface rover missions with constructive simulations of a multitude of mission specifics including autonomous navigation. DARPA invested in the open source Gazebo environment to create the DARPA Robotic Challenge Simulation (DRCSim) system [7]. DRCSim is a collection of models, environments, plugins, and tools that customize the Gazebo simulator for use in the DARPA Robotics Challenge (DRC). Gazebo contains general simulation capabilities and DRCSim includes the robots, objects, and code that are specific to the DRC.

TARDECs Analytics area has also explored AVPG development with a focus on inclusion of high fidelity vehicle dynamic models [8, 9]. The focus here has been on trying to get the simulation environments to run in real or near real-time with more representative models of vehicles.

Finally there has been many other APVG like systems developed as freeware [10] and there have also been systems developed by individual UGV autonomy developers to test and improve their product. As in the case with the DRCSim and Webots many of these virtual environments have been transitioned into systems that are now being used to host online competitions as well as sharing as freeware to the public. In fact, offerings such as Webots have converted from freeware provided to the public, via GNU license agreement, to commercial offerings due to the rise in popularity of the environment. This popularity rise can at least in part be traced back to use of these systems in the online competitions.

APVG Components

To make a statement on what facets must exists in an APVG we first need to set the expectation of what the intended end-user of the APVG would use the product for. For the purposes of this paper we are restricting our APVG description to one that will be used by developers to test their autonomy and control modules in a holistic fashion prior to implementing the code on the actual vehicle. The important features of such a system can be broken into a few key subgroups and configured to work together usually in a method similar to the one shown in Figure 1. It is important to note that in Figure 1 the Sensor Model is meant to represent perception sensors and the Vehicle Model is meant to self-contain any “self” information that would come from proprioceptive and state sensors.

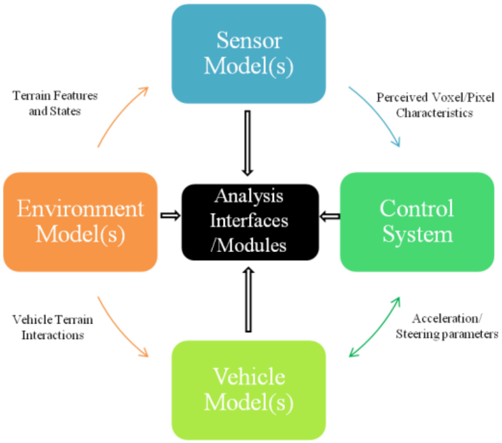


Figure 1: Common Modules of an AVPG

Environment Model(s): This component of any AVPG layout, such as presented in Figure 1, is core to the entire process. This is the component which provides the primary stimulus to many, if not all, of the other modules of the AVPG. At a minimum your sensor/perception model and your vehicle model will interact with the environment model to produce stimulus to each of these systems. The level of detail contained in this module varies per development environment. Of the simulation environments detailed in this paper VANE provides for the highest fidelity with respect to the environment. Depending on the task specifics VANE will sacrifice computation time in order to provide the representation of the terrain in detail. Of the systems to this point Stage is the most simple in terms of environmental fidelity. It provides for a 2-Dimenisional representation of terrain without complex reflectance/refraction and material representations. Figure 2 depicts a sample Stage environment with a robot vs. a VANE environment with robot (shown in contrast to the real environment image).

The level of fidelity required in an environment model is really a function of the purpose of the robotic simulation. If the purpose is for testing of perception algorithms with a complex sensor model than the environment model should be high fidelity. If the purpose of the simulation is for testing of vehicle control and planning algorithms the level of fidelity of the environment may not be as important depending on if the algorithms are affected by terrain variations.

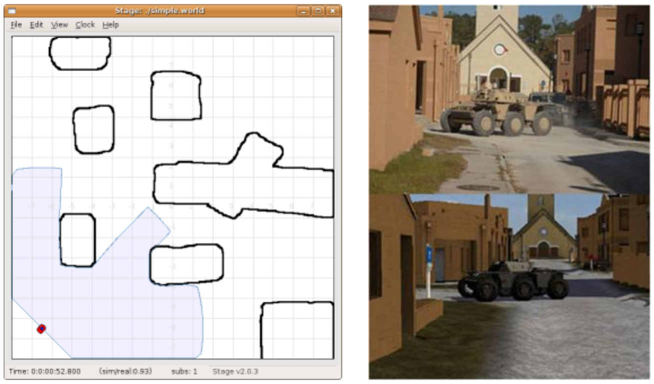


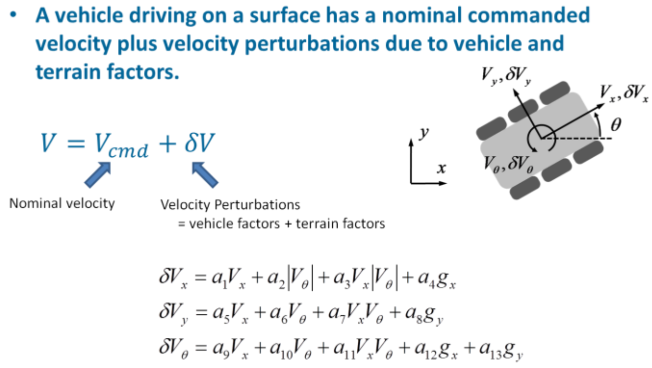
Figure 2: a) Stage environment vs. b) VANE environment

Sensor Model(s): As with the environmental model the required fidelity of the sensor model is dependent on the type of task being explored. If the purpose is to develop and test Obstacle Detection/Obstacle Avoidance algorithms without specific requirements on the fidelity of sensor than a Stage sensor model (shown in Figure 2a) may be appropriate. If the purpose is to determine a clustering algorithms stability/suitability given sensor intrinsic properties than a high fidelity sensor model would be required (as well as a likely high fidelity environment model). The systems described in this paper that are capable of high fidelity sensor modeling, in their current configurations, are VANE and MODSIM. Both of these systems are designed to work on high end machines for this very purpose.

An important point to make about the sensor model in any AVPG is that ideally the interface drivers to the real sensors are also used to control the virtual ones as well. In this way we are making the communication between the control system and the sensor module independent of the actual hardware being in place (hardware abstraction). This is important feature to reduce the amount of effort required to use the AVPG.

Vehicle Model(s): The vehicle model is the plant of the simulation. It is the item that is receiving control commands from the control system module and being excited by the input from the environmental module. The most common starting point for vehicle models in these simulation environments is the Open Dynamics Engine (ODE) [11]. ODE is used for simulating the dynamic interactions between bodies in space. It is comprised of a rigid body dynamics engine and a collision detection engine. The ODE wiki [12] provides a tutorial on how to implement a 4 wheel vehicle and that is utilized by most simulation environments as a basis for their underlying vehicle model. ODE has some drawbacks that are being looked at [13] regarding the method of approximating friction and poor support for jointdamping. MODSIM has its own library of vehicle models and library of Vehicle Terrain Interactions (VTI) for a multitude of vehicle styles.

The work by the TARDEC's analytics group has been to try and put more complex vehicle models in-line with these autonomy simulation environments [8, 9]. To this point they have used simpler representations of the environmental and sensor models to try and meet the high-fidelity vehicle model goal in near-real time. The goal has been to try and achieve real-time performance on commercial machines. As the level of fidelity of the vehicle model goes up so will the level of required fidelity of the environment to achieve relevant and realistic vehicle terrain interactions. This is especially true for the smaller UGV platforms. The National Robotic Engineering Center (NREC) is currently working on developing a real time ultra-high fidelity virtual environment for autonomy testing based on real sensor data [14]. An example of the type of VTI equations that could be considered with a high fidelity vehicle and high fidelity terrain model can been seen in Figure 3.



- A vehicle driving on a surface has a nominal commanded velocity plus velocity perturbations due to vehicle and terrain factors.

$$V = V_{cmd} + \delta V$$

$$\delta V_x = a_1 V_x + a_2 |V_\theta| + a_3 V_x |V_\theta| + a_4 g_x$$
$$\delta V_y = a_5 V_x + a_6 V_\theta + a_7 V_x V_\theta + a_8 g_y$$
$$\delta V_\theta = a_9 V_x + a_{10} V_\theta + a_{11} V_x V_\theta + a_{12} g_x + a_{13} g_y$$

Figure 3: Example displaying how high fidelity vehicle and environmental models can be used to identify the effects of VTI perturbations in a skid steer wheeled vehicle.

Control System: The control system module is the location that the end user of an AVPG should spend the vast majority of their time. Ideally the entire AVPG would act as a hardware abstraction layer to the vehicle and test environment. An example of an attempt at this type of scenario can be seen in Figure 4. In Figure 4 we give an example of TARDECs Dismounted Soldier Autonomy Tools (DSAT) program utilizing the ANVEL AVPG. Here the simulation interacts with the Hardware in the Loop (HIL) Platform Controller (tablet or phone) to control the virtual vehicle model (Polaris MRZR) in the virtual environment (ANVEL) using the real vehicle control software (Robotic Technology Kernel (RTK)) with a single conversion module (ANVEL2ROS) to allow for the hardware and terrain abstraction.

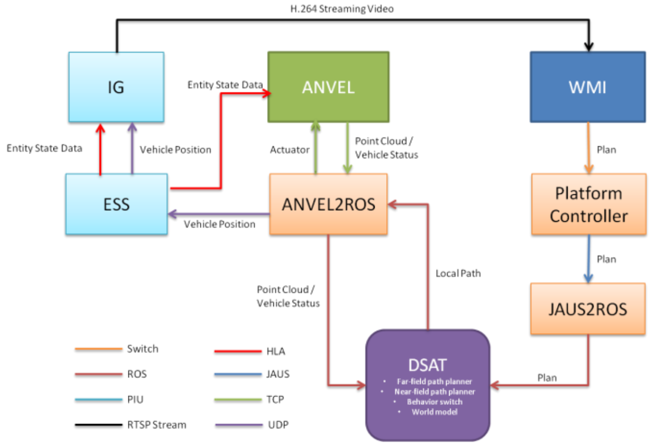


Figure 4: Example of DSAT control software working in the ANVEL Plug-in AVPG. Here the user interacts with the Warfighter Machine Interface (WMI) software installed on the Platform Controller (Tablet/Phone) to interface with real DSAT RTK control software to control the virtual Polaris MRZR in the ANVEL generated virtual terrain. TARDECs Embedded Simulation System (ESS) interacts with the ANVEL virtual terrain to produce the imagery (IG) sent to HIL Platform controller in the same fashion the real vehicle would send information to the WMI software for display to the user (scene/camera imagery for teleop or autonomy modes).

Analysis Interface: This component of the AVPG is at the very core of why there is such a desire for these systems in the autonomy development market. It is a very time consuming and costly endeavor to implement robotic control algorithms onto platforms and test these platforms in the real world every time a developer wants to test a new algorithm at the system level. It is much easier for that developer to setup his test conditions in an AVPG and perform as many runs as necessary to collect the desired data to assess the value of this modified, or new, set of code. This is the very reason AVPGs exist at the developer level and the way in which data is collected from each of the components of the system is important. Most of the AVPGs allow you collect data in the format you prefer (for post analysis). Some of them have methods to port data directly to Matlab and Labview environments. A few (ModSim and ANVEL) have embedded methods to analyze data during simulation execution, even allowing the developer to alter parameters on the fly as well.

AVPG Implementations

In this section we will give an overview of the AVPG implementations we have performed in the GVR area of TARDEC over the past 5 plus years. These implementations range from use on TARDEC In-Lab In-house Research (ILIR) efforts, use in competitions, use on Army Technology Objectives (ATO) programs, use on a Depart of Defense (DOD) Joint Capability Technology Demonstrator (JCTD), and finally use on a quick response fielded autonomy program for a Special Forces unit. These implementations have varied in level of use during development, level of fidelity required of component modules and level of integration with the respective program effort. The AVPGs that will be discussed include Player/Stage/Gazebo, ModSim, ANVEL/VANE.

Player/Stage/Gazebo: We have had a few implementations of these AVPGs in the past couple of years due to the availability and ease of use of these solutions. For these reasons we have used this AVPG for ILIR efforts and others have used it as parts of competitions that we have sponsored and administered.

Player/Stage: First we have used Player/Stage on a few ILIR efforts as a simulation and development platform for several mapping and navigation algorithms. Player/Stage is an open-source robotics simulation tool used to model intelligent systems and is designed to minimize the overhead process of simulating the behavior of a robot so that the designer can focus on developing the controls and intelligence of the system. Player/Stage provides a way to quickly model a robot, sensors, and environment for the simulation while also giving the user complete control over all aspects of the simulation. Player is the low level hardware abstraction layer and contains the drivers for the hardware and interfaces the controller to the hardware. Stage is a simulator that player plugs into. Stage simulates data that is given to the sensors and the movement of the robot in an environment.

We used Player/Stage to test two separate robotic control approaches (potential fields and formation control) when assumptions regarding a basic understanding concerning the bounding dimensions of terrain to be negotiated (for potential fields) and a common level of cognition between multiple robotic controllers (for formation control) are not upheld in the design criteria or mission execution. First a modified potential-field algorithm was developed/tested for singular robotic navigation in cluttered, unconstrained dynamic environments and then an algorithm for reactive formation control was modified to explore the stability of the solution when humanistic variability is introduced to the system (Figure 5). In both cases the Player/Stage AVPG served as a suitable testing environment as the purposes of the use of the AVPG was only to extract information regarding the suitability of the pure control approach guiding the robotic navigation rather than independent or complex perception or dynamic variables.

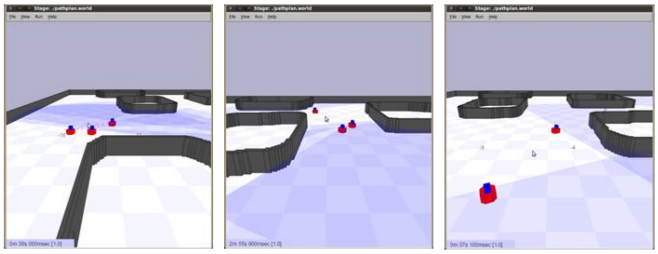


Figure 5: Example of robotic entities following a human controlled lead entity and failing to maintain track of the lead entity when encountering terrain occlusions. In the first image two subservient followers are following the human driven leader. In the second image one entity has broken from the formation while the other is still following the human leader. In the third image both followers have broken from following the human driven leader.

In addition to the testing reactive robotic control methods described in the previous paragraph we have also used the Player/Stage AVPG to test the suitability of various deliberate heuristic based planning techniques in varying size and varying obstacle environments. We developed a Player/Stage set of procedures that allows us to implement heuristic planning algorithms onto a preset selection of virtual vehicles and sensors. We are able to place these vehicles into a variety of 2 dimensional environments and select, or develop, a new planar of choice (A*, D* and D*Lite were already integrated into the system by us) and then give one or more of these vehicles a path to execute while recording the coma delimited data from each (Figure 6). Again, given the focus on the vehicle control algorithms independent of terrain and perception sensors we are able to execute these tests in the Player/Stage of AVPG.

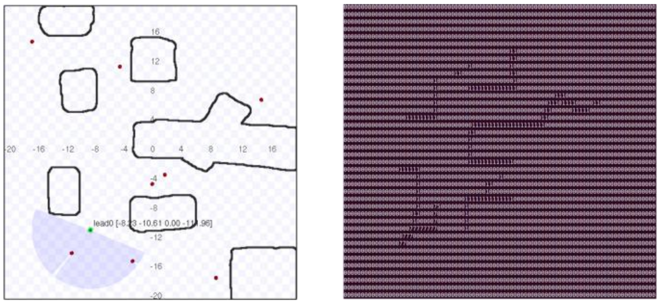


Figure 6: A single pioneer robot (green) navigating to a preset location using GPS and LIDAR sensors. It started in the top right corner and had no previous knowledge of its environment. The map generated by the pioneer as it traversed the map. 1's represent walls, 0's are empty space or unknown area, and 7's represent the planned path for the robot to take using the A* path planning algorithm.

Gazebo: Gazebo is the basis for the DARPA Robotics Challenge (DRC) Simulator, DRCSim. Participants in the DRC program used the DRCSim during the Virtual Robotic Challenge (VRC) portion of the down selection process, a cloud based simulation competition held in 2013. The DRCSim extends Gazebo with a collection of robot models, robot components, and field environments that are specific to the DRC. The (Figure 7) is the first phase of the DARPA Robotics Challenge (DRC) which provides teams the opportunity to compete via the Cloud for funding to the next phase involving implementation of code on real robotic platforms.

The next phase, called the DRC Trials (DRT), was held only several months later on using real hardware, which by design forced teams to use simulation for early development and testing of their algorithms. The VRC competition consisted of a human-in-the-loop simulation to control an Atlas robot to perform three tasks with five stages in each task. The tasks stretch the state of the art in legged robot control, mobility, manipulation, and driving skills. Each stage of each task is performed with variations in the environment (e.g. Driver's steering wheel is larger/smaller, more friction on pedals, etc.). Although there were increases in the number of challenges in the DRC Trials, it was shown that five of the top eight teams in the VRC were among the top 8 teams in the DRT.

Figure 7: Example of Gazebo rendering of DRC robotic Atlas humanoid system, control vehicle and other terrain features

DARPA provided a great deal of funding to the Open Source Robotics Foundation (OSRF), the administrators of Gazebo, to make enhancements which benefit the Robotics community and the VRC. Their ultimate goal for these enhancements is to allow labs to compare results, share code, and reduce barriers of entry into the Robotics community by allowing robotics research work to continue without the need of physical systems or geographical locations. These enhancements include the development of CloudSim, used for control simulations in the Cloud, increased environmental models, GUI robotic controls, Physics engine support (includes ODE, Bullet, Simbody, and Dart), and plug-in APIs for specific control of environment and components.

Gazebo is also used by teams in the annual International Ground Vehicle Competition (IGVC) each year to test their entries on expected course layouts and obstacle layouts (Figure 8).

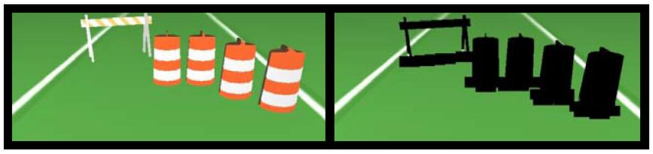


Figure 8: Example of Gazebo rendering IGVC obstacles both from the perspective of the human observer and how these objects are represented to the robot.

ModSim: ModSim is a **Mod**eling and **Sim**ulation environment supporting the engineering, design, development and testing of autonomous navigation systems. It is a high fidelity environmental and vehicle, real-time, simulation environment with embedded analysis tools. It was first developed for the Future Combat System autonomy system but has since transitioned to TARDEC where it is currently being used on the DOD Autonomous Mobility Appliqué System (AMAS) JCTD. The system has since been renamed the Robotic Intelligence Development Environment (RIDE) and has been modified to accept multiple versions of the AMAS autonomy control system. The simulation has a very large selection of high fidelity vehicle, terrain and sensor models. This system also allows for constructive and batch simulations and is designed to be operated in a model-test-model format where the performance of the AVPG can be improved after real test runs (Figure 9).

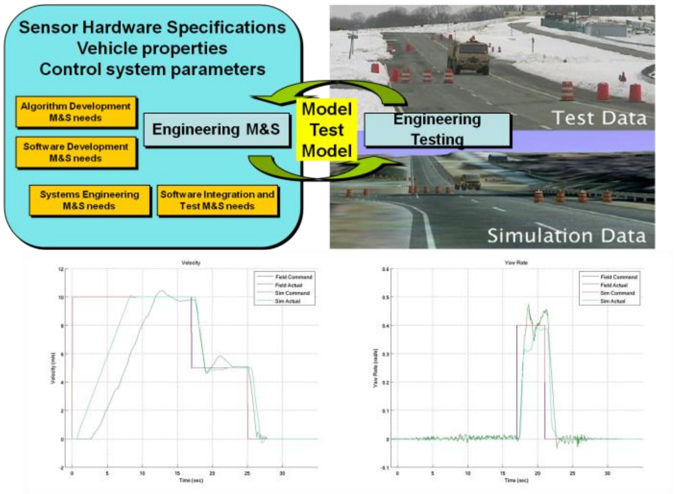


Figure 9: The MODSIM Model Test Model paradigm allows for constantly increasing model fidelity level as shown in the analysis of the FMTV actual and simulated control response curves. Where significant response lags were found in controller reaction time between the real vehicle and how the vehicle was acting in simulation. The parameters in the simulation were subsequently updated to reflect and correct this discrepancy.

<u>ANVEL/VANE</u>: ANVEL is an easy to use simulation tool that allows for testing of basic behavior and functionality when high fidelity simulation is not required. ANVEL contains easy to use tools for generating terrain and placing obstacles in a scene and is also capable of loading Ogre3d compatible mesh models that have been developed using other software.

One primary way in which ANVEL is used in the Dismounted Soldier Autonomy Tools (DSAT) program is to test the Warfighter Machine Interface (WMI) software (Figure 10). This is the controller software which the vehicle operator uses to select autonomous behaviors manually tele-operate, manage waypoint plans, and monitor vehicle status. This level of testing generally involves transitioning between various modes and seeing that the system responds correctly. This phase of testing relies on having realistic sensor health and status information, but does not generally benefit from having a highly detailed world model.

Figure 10: Example of DSAT HIL interfacing with platform controllers and the ANVEL AVPG

Another use for ANVEL is testing autonomous vehicle behaviors. The main behaviors tested for DSAT include tele-operation, waypoint following, and dismounted soldier following. Both tele-operation and waypoint following are straightforward to perform. The ANVEL system populates the drive camera with a video stream of the virtual environment, and provides a point cloud to substitute data from the LADAR sensor driver which is used by DSAT for obstacle avoidance. Dismounted soldier following requires some extra configuration. In addition to the terrain and obstacles, the user must define a waypoint routine for the dismounted soldiers to execute. Currently, these waypoint routines are defined and executed from a separate tool (ESS) which is then played back at runtime to control and animate a dismounted soldier within ANVEL.

In order for the DSAT and WMI software to communicate with ANVEL, a Robot Operating System (ROS) module was made (ANVEL2ROS, see Figure 4). This module receives data from ANVEL based on their limited Application Programming Interface (API) and reformats it into ROS messages that the DSAT software would expect to come from the actual sensors. Using a simple network socket, the ROS module sends drive commands to ANVEL which would normally be sent to the vehicle's drive-by-wire system. Using the same network socket, ANVEL sends vehicle position and velocity, LADAR sensor data, and camera data back to the ROS module. In order to match the output of the real DSAT LADAR driver, which provides whole point clouds instead of single scans, the ANVEL ROS module must buffer the points it receives until it has an entire point cloud, as well as interpolating between points to match the resolution that DSAT expects.

Finally, TARDEC is utilizing ANVEL to perform a sensor height study to determine whether raising the height of the LADAR sensors on the DSAT system helps to improve performance of autonomous waypoint following. By raising the LADAR sensors higher up, the theory is that this should increase the density of measurements taken further away from the vehicle. If that proves to be the case, this should help to improve the distance at which DSAT can identify obstacles and provide ample time for the vehicle to avoid them. This is vital to being able to perform autonomous behaviors at higher speeds. To test this, a set of obstacle courses were made with varying levels of difficulty (Figure 11). The vehicle was then setup to perform the same waypoint plan through each of the courses with the LADAR sensors set at various heights. Once the data has been collected for each combination of course and sensor height, the evaluation criteria used will be based on the time it takes to complete the course and how often user intervention is required.

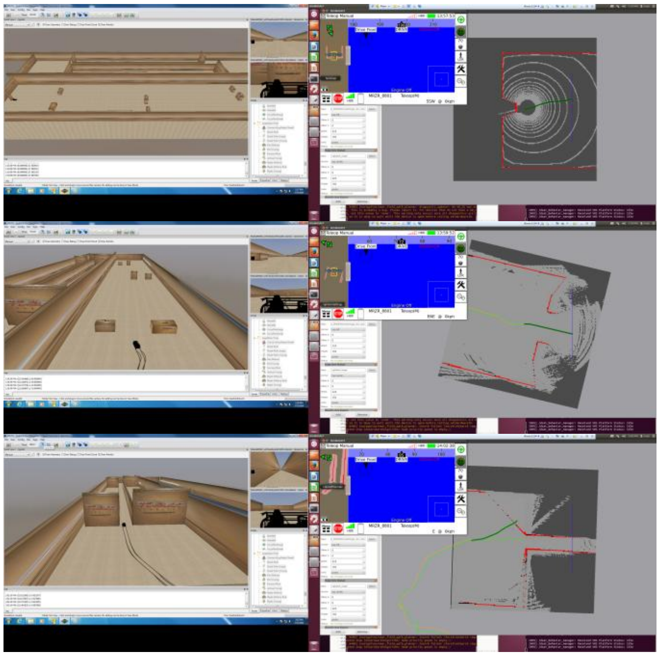


Figure 10: Examples of the DSAT AVPG based height study test course and select image captured runs

TARDEC and ERDC have used VANE on an ATO to feed autonomy safety measures in complex urban environments (Figure 2). The simulation utilized the high fidelity nature of VANE to determine the UGVs path through a maze of Constantine wire, which could not be picked up by the sensor models in the lower fidelity ANVEL simulation. This was done to demonstrate that there are situations where a solution may not be able to be determined with real-time simulations as the fidelity of the environment and sensor models are not likely to be accurate enough to represent such thin objects in space.

Conclusions

There is not likely one AVPG solution that is going to be created that is going to solve the needs of everyone in the community. This paper only detailed the interest in AVPG from a developer's perspective and even in that select domain we have demonstrated cases where different facets of the task at hand dictated a different AVPG solution for a variety of reasons (fidelity requirements, ease of use, availability, sponsorship and investment by an outside organization). It is the suggestion of the authors of this paper that AVPG solutions continue to be developed, invested in, investigated and utilized for applications including those outside the domain of just development. These systems have the potential to bridge the gap in many areas for robotics (many of which we have already been covered in this paper a few times). It is also the belief of the authors that continued utilization, and development, of both open source and government/commercial solutions will be necessary in order to get the robotic community to the point where the testing, understanding, reliability, trust and knowledge of how these systems are intended to act is sufficient enough for autonomous robotic assets to become the force multipliers that they are envisioned to be by so many.

REFERENCES

[1] B. Gerkey., R. Vaughan,, A. Howard (2003). *"The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems."* Proceedings of the International Conference on Advanced Robotics 317-323

[2] O. Michel (2004). "*Webots: Professional Mobile Robot Simulation*", International Journal of Advanced Robotic Systems", Vol. 1, Num. 1, pages 39-42.

[3] G. Kreafle. "*Advances in Gaming and Simulation Applied to Robotics"*, Joint Center for Robotics: 5th Robotics Workshop, June 19, 2009

[4] P. Durst, C. Goodin, C. Cummins, et. el., *"A Real-Time, Interactive Simulation Environment for Unmanned Ground Vehicles: The Autonomous Navigation Virtual Environment Laboratory (ANVEL),"* icic, pp.7-10, 2012 Fifth International Conference on Information and Computing Science, 2012

[5] P. Rander, B. Browning, *"'Empirical Evaluation of the Virtual Autonomous Navigation Environment"* Final Report to the Army Research Office, 2009.

[6] A. Jain, J. Guineau, C. Lim, et. el., *"Roams: Planetary Surface Rover Simulation Environment"*, International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS2003), (Nara, Japan), May 19-23, 2003.

[7] DRCSim: http://gazebosim.org/

[8] P. Jayakumar, W. Smith, B. A. Ross, et.el. *"Development of HighFidelity Mobility Simulation of an Autonomous Vehicle in an Off-Road Scenario Using Integrated Sensor, Controller, and Multi-Body Dynamics"* in 2011 NDIA Ground Vehicle Systems Engineering and Technology Symposium, Dearborn, Michigan, 2011.

[9] J. Cameron, S. Myint, P. Jayakumar, et.el. "*REAL-TIME AND HIGH-FIDELITY SIMULATION ENVIRONMENT FOR AUTONOMOUS GROUND VEHICLE DYNAMICS*" 2013 NDIA Ground Vehicle Systems Engineering and Technology Symposium, Troy, Michigan 2013

[10] Sims: http://en.wikipedia.org/wiki/Robotics_simulator

[11] Open Dynamics Engine (ODE): http://www.ode.org

[12] ODE Wiki: http://ode-wiki.org/wiki

[13] Drumwright, et al. (2010). *"Extending Open Dynamics Engine for Robotics Simulation"*. Simulation, Modeling, and Programming for Autonomous Robots. Springer.

[14] http://www.nrec.ri.cmu.edu/about/news/11_12_tools.php