PVS: A System For Large Scale Outdoor Perception Performance Evaluation

Cristian Dima[1], Levi Lister[1], Joan Campoy[1], Carl Wellington[1], Carlos Vallespi[1], Boyoon Jung[2], Michio Kise[2], Zachary Bonefas[2] and Stewart Moorehead[2]

***Abstract*— This paper describes the motivation, design and implementation of the Perception Validation System (PVS), a system for measuring the performance of outdoor perception systems for autonomous vehicles. The system relies on using large amounts of real world data and ground truth information along with customizable metrics in order to quantify performance aspects such as the rate of false positive or negative detections of an obstacle detection system. Our system relies on a relational database infrastructure to achieve a high degree of flexibility in the type of analyses it can support.**

We discuss the main steps required for going from a raw data log of the data produced by the sensors mounted on a vehicle to numerical estimates describing the performance of the perception system, including the generation of ground truth information and one performance metric which we found to be very useful for comparing the perception system's outputs to the ground truth.

Several examples of the type of information that can be obtained using the Perception Validation System are presented.

I. INTRODUCTION

A. Motivation

Significant progress in the area of outdoor mobile robotics has brought real world applications of autonomous navigation in military, agricultural, mining and transportation domains closer to reality than ever before. In this context, the ability of an autonomous vehicle to perceive and understand the environment in which it needs to operate remains one of the main obstacles on the path to commercialization: without a capable and reliable perception system the ability of a mobile robot to safely and efficiently execute tasks is severely limited.

The robotics community has attacked this obstacle vigorously, deploying an impressive array of techniques and technologies. It is now almost common knowledge that obtaining good perception performance in unstructured outdoor environments requires at least some reliance on machine learning based approaches. We also know that the need to build systems that operate robustly despite significant changes in their environment often leads to perception systems relying on several different sensing modalities and data processing algorithms. While these types of approaches led to significant capability improvements in this area, the complexity of the perception systems being developed has also increased significantly.

1: National Robotics Engineering Center, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15201, USA. Corresponding author: cristian.dima@gmail.com

2: John Deere, Moline, Illinois, USA

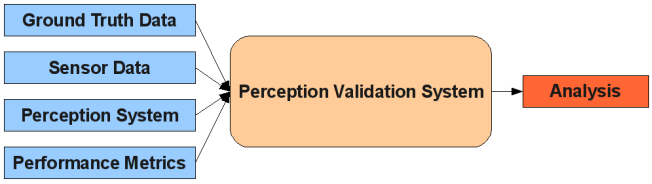


Fig. 1. A high level view of the PVS system. The system relies on processing a large amount of pre-recorded raw sensor data with a perception system, and comparing its outputs with a corresponding set of ground truth data using configurable performance metrics. A database system manages the entire process, and supports the necessary tools for performing detailed performance analyses.

The importance of the perception performance levels coupled with the seemingly inescapable complexity of the solutions developed in order to maximize them lead to a fundamental question that needs to be answered before any large scale commercial applications can be considered: how can one *understand, describe and make justified guarantees* –even in a probabilistic sense– about the performance of a complex perception system that is critical to the safe operation of a robot?

Our belief is that any satisfactory solution to this fundamental question must rely at its core on the ability to *measure performance* in a relevant and trustworthy manner. We contend that in our context, a performance measurement is rendered relevant and trustworthy by the following characteristics:

- The performance metric being employed must be linked as directly as possible to the actual purpose of the perception system. As an example, a perception system designed for avoiding collisions with trees should be measured primarily by its ability to prevent such collisions while also measuring its impact on the overall productivity of the larger robotic system.
- The raw data on which the measurements are based must be collected in environments and conditions that are directly relevant to the application of interest, and the amount of data used must be *sufficient* for the purpose of the analysis: it needs to encompass all the major environmental variations that can be expected over the lifetime of the system being developed. As a rule of thumb, we consider a data set to be sufficiently large and representative if it contains several examples of the situations the perception system is expected to function correctly in, under all major environmental variations. Unless this data is included in the tests,

one is likely to make wrong assumptions about the generalization properties of the perception system.

This paper describes a system for quantifying the performance level of the perception subsystem of autonomous mobile robots. In addition to facilitating the understanding of the capabilities of the perception system, we have designed the system so that it is also a useful tool for accelerating perception research and development. We will introduce the three main usage models for the *Perception Validation System* (PVS) by describing the problems it is designed to solve and the type of questions that it can help answer:

1) *Determine perception system capabilities*. This ability is important for understanding the mapping between capabilities and potential applications. Potential users might be test engineers or marketing experts, who might be interested to determine if a given system is capable enough to support applications that require driving through a farm, an orchard or corn field, a road, surface mines or other environments of interests. Test engineers could use the system to efficiently verify that a given perception system meets the system requirements. For example, they can verify performance in challenging lighting conditions (dawn, dusk or night time) or in the presence of air obscurants (dust, rain, smoke or snow) or compressible ground vegetation such as weeds.
2) *Compare multiple perception systems*. Measuring the performance values obtained when using different sensor combinations, amounts of computing resources, algorithms or parameters for those algorithms with relatively low effort is a tremendously useful capability for the design and development phases of building a perception system. A developer can make design decisions based on hard data regarding their impact on the performance of the overall system, instead of relying on intuition. For example, the PVS system enables one to *measure* if for a particular application using a laser range finder is more effective than using stereo vision, if using a higher resolution camera is justified by performance improvements or if using lower powered but more rugged computing hardware has any impact on perception performance. Similarly, one can measure the impact of using a new algorithm, or changing a particular parameter value from A to B.
3) Since the PVS measures perception performance at the highest level, it can also be used to *measure the capabilities of a entire perception system at a given moment in time*. This enables it to be used as a system-level regression tool that can capture the effects of the often non-trivial interactions that arise in complex perception systems designed for real world environments. A manager can use such regression tools for tracking development progress over the course of a project. A development team might track the exact same values in order to detect software changes that actually result in a degradation of the overall system performance.

B. Related Work

The idea of measuring perception performance has been around for many years, even in the relatively narrow domain of outdoor autonomous navigation. We can conceptually group the previous efforts in the area of perception performance evaluation into system-level and low-level approaches, based on the level at which the measurements are done.

The system level approaches depend on methods for measuring performance at the system level, and have been widely promoted by DARPA[1]. Between 2000 and 2003, the DARPA sponsored PerceptOR program ([1]) used system level Evaluation Experiments (EEs) in challenging real world environments to differentiate several teams participating in the program. The quantitative metrics employed were based on testing in *unrehearsed* environments. A government testing team defined obstacle courses by selecting a series of waypoints provided to the participating teams by their GPS coordinates. A performance baseline was established by having an expert human driver drive the courses on an ATV platform similar to those on which the autonomous vehicles were based. A second baseline was established by having a remote operator teleoperate the same ATV platform based on video streaming at 2.5Mb/s from a vehicle-mounted camera. During robot testing, the testing team monitored variables such as the distance traveled by the robot, the duration of the traverse, the median speed, the number of remote operators supporting the traverse, the uplink and downlink data volumes (as an estimate of the amount of autonomy vs. teleoperation), the uplink and download data volumes per meter traveled, the number of emergency stops and failed traverses. Over the four years of the program, the autonomous vehicles performed 296 runs, totaling a traveled distance of 130km and a navigation time of 110hr. The testing team was able to aggregate numerous metrics such as the number of emergency stops per km (reduced by a factor of 22 during the course of the program) and the uplink data volume per unit distance (reduced by a factor of 46 over the same period of time), demonstrating the value the program for accelerating technology development.

A somewhat similar approach was adopted for DARPA's Learning Applied to Ground Robots (LAGR) program which occurred between 2005 and 2008 ([2], [3]), as a follow-up on PerceptOR geared specifically towards machine learning approaches applied to autonomous navigation. For this program (also structured as a competition), all the participating teams had to use identical vehicles, and all the testing was performed independently by a government team without any members of the competing teams being involved in the tests. A great overview of the LAGR program is provided in [2].

DARPA took the multi-team competition to an extreme during its two off-road "Grange Challenge" events in March 2004 and October 2005 and the "Urban Grand Challenge"

[1]The United States Defense Advanced Research Projects Agency)

event in November 2007 (see [4]). These events were all structured as races, in which all teams had to compete over the same course with the only performance metric being the time required to complete the course while respecting a set of rules primarily designed for the safety of the race. The trade-offs related to choosing particular algorithms, sensing modalities or race strategies were primarily presented in post-race publications by the participating teams based on independent testing or on processing data logs recorded during the race (see [5], [6], [7], [8], [9]).

A final example of robotics autonomy programs which measured performance at the system level comes from DARPA's UGCV-PerceptOR Integration (UPI) program, designed to achieve a leap in autonomous navigation performance by combining an advanced off-road vehicle platform with novel approaches to the design of perception and vehicle control software. Similarly to PerceptOR, the field tests consisted in performing autonomous traverses following series of waypoints spaced between 0.2-2km in terrains varying from temperate forests to high deserts and presenting obstacles such as high slopes, large boulders and dense vegetation (see [10]).

The program followed aggregate performance measures such as the average travel speed over the course, the number of operator interventions per distance traveled and the total distance traveled. Effort was also dedicated to understanding the impact of using different perception algorithms on the performance of the overall system. Autonomous runs were performed repeatedly using prior data at different resolutions, and using different configurations of the main components of the autonomy software. In [10], the authors define additional metrics in order to document the improvements in navigation performance obtained as a result of applying different machine learning based approaches.

These competitive programs centered around testing in challenging world environments pushed the autonomous navigation capabilities to the limit and beyond, and the efforts towards measuring performance in an objective manner played a major role in their success. Unfortunately, the exorbitant costs of testing on real robots in different environments put a severe limit on the amount of information that can be extracted by only measuring performance at the system level. Each system configuration to be tested required an independent run over the entire course. The UPI program was perhaps the most advanced in this respect, and yet it was quite limited with respect to the number of configurations that were tested on the vehicle, and the amount of experimentation in different environmental conditions.

An opposite alternative to the system level testing approach is to use standard techniques from the statistical machine learning and literature to compare the outputs of the different components of a perception system to ground truth data. Typical performance measurements in this area rely on false positive or false negative detection rates, receiver operating characteristic (ROC) curves, precision/recall curves, confusion matrices and "area under the –ROC– curve" (AUC) measurements (see [11] or any standard texts on statistical learning). The use of such metrics is relatively simple and it makes intuitive sense: one presents the system with pre-labeled examples of data, and then estimated the probability of making different types of mistakes. Although the assumptions required are strong –theoretically the test data needs to come from the same distribution as the data from the real environment where the perception system is to be used–, these measures have become a standard not in only inside the machine learning field but also in robotics and computer vision applications.

While flexible in terms of the analyses that can be performed since each component can be tested separately, low-level testing has significant limitations of such related to the difficulty of producing large amounts of labeled data that is meaningful for the perception tasks to be performed. As an example, in an obstacle detection application we not only want to know the false positive and false negative rates expected for system based on all of our data, but we want to understand what those rates are for different types of obstacles located at different ranges, in different weather conditions, in different locations relative to the vehicle, etc. For a vision based perception system, having an expert label important images in a dataset and then comparing the outputs of the system to the labels is relatively easy –though still expensive–, but it not easily capture any of the context aspects discussed above. Another challenge for these standard tools is that the in reality, the penalties that apply to perception errors are highly non-uniform: committing perception errors very far the vehicle should ideally have less impact on the overall perception numbers than errors commited close to the vehicle where the decisions made by the perception system are critical. It is hard to capture such non-uniformities using just the standard measures described above.

The system we describe in this paper is meant to fill the gap between the system level testing (objective, measuring critical end performance but extremely expensive and hence typically rough in the type of information extracted) and low-level component testing (flexible in terms of the analyses that can be performed, but challenging in terms of assuring that the right data distributions are used and the right conclusions are drawn). As indicated on Figure 1, the system relies on processing a large amount of pre-recorded raw sensor data with a perception system, and comparing its outputs –generally two dimensional output maps describing the environment around the vehicle– with a corresponding set of ground truth data using configurable performance metrics. A database system manages the entire process, and supports the necessary tools for performing detailed performance analyses.

A somewhat similar approach was recently and independently proposed in the meta-learning area of machine learning by [12]. The author describes a large scale system for accumulating data sets, algorithms and parameters from the current machine learning literature, and using it to make discoveries about the behavior of different learners. In addition to this, the authors convincingly argue that having

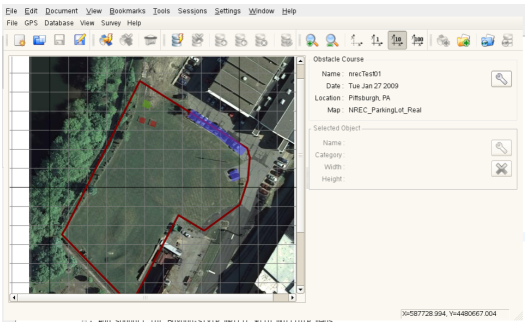


Fig. 2. User interface used for using a GPS survey to obstain ground truth information for an obstacle course.

such a system available will improve the repeatability of the experimental results presented in the literature, and protect against a false sense of progress that can result from testing new algorithms on relatively small sets of data.

To summerize this section, we have discussed the motivation for developing the PVS system which enables the efficient and detailed exploration/verification of perception capabilities critical for commercial applications of outdoor mobile robotics. We have also argued that being able to easily perform large scale experiments with different variants of a perception system can lead to a significant acceleration of research and development work, and that current performance estimation tools –whether at the system level or at the component level– fall short of providing the necessary capabilities.

In Section II we will discuss the requirements that were considered during the design phase of our system, and in Section III its implementation. Sample analysis results obtained with the PVS are presented in Section IV, our conclusions and future work are presented in Section V.

II. DESIGN REQUIREMENTS

In the introduction we have hinted that two important characteristics of the validation system must be the ease and reasonable cost of collecting large amounts of raw and ground truth data, and the ease of use and flexibility of the analysis tools and the infrastructure used for measuring performance. In this section we present in more detail the main design requirements in these two areas of the system.

A. Data Collection Requirements

- Collecting raw data needs to be fast and simple. For the case of a vehicle which can be both manually and computer-driven, the process should consist of simply having an operator traverse the environment of interest.
- Collecting ground truth data should be cost-effective in terms of the effort required for labeling large amounts of data.
- For a system with limited hardware changes, it should be possible to reuse old data logs such that a large body of real-world data is accumulated over time.

B. Analysis Requirements

- The PVS should support multiple systems, with variations in vehicle platforms, hardware and software configurations, operating environments and intended applications.
- One should be able to perform analyses in an intuitive manner. Users without perception or programming expertise should be able to use the PVS and understand the capabilities of the system being analyzed at the high level. Examples of such potential user groups are quality assurance testing engineers or marketing experts.
- The analysis system should offer very fine control of the configuration of the systems being tested, in order to support the low level analysis needs of the development group. One should be able to measure the impact of very fine changes deep inside the perception system being analyzed.
- Offer very fine control over the set of data logs that are used to measure the performance of the perception system, in order to support analyses focused on different environmental conditions (e.g. illumination, presence or absence of rain or dust), different types of obstacles (e.g. vehicles, people, trees, buildings) and different operating environments (e.g. open fields, roads, orchards).
- The analysis system needs to support processing massive amounts of data, which are typically required in order to draw reliable conclusions regarding the performance in highly variable outdoor environments.
- Since it is virtually impossible to anticipate all the types of analyses or that one might want to use the PVS system for, it should be possible to easily change the types of questions one attempts to answer based on the PVS, or the performance metrics.

Collecting huge amounts of data while obtaining ground truth in a cost effective manner, supporting multiple different perception systems and also future types of analyses while at the same time having a user interface that is non-expert friendly yet powerful enough for developers is challenging. The next section will present the implementation of the PVS system, which we believe satisfies the majority of the system requirements.

III. IMPLEMENTATION

A. Raw and Ground Truth Data Collection

Our raw data collection process relies on recording almost all the stream of data produced by the sensors in a perception system: positioning sensors such as IMUs, GPS units and wheel encoders, perception sensors such as laser range finders and video camera and information produced by the vehicle platform. In addition to this, with each data log we store meta-data such as the location where the data is recorded and the atmospheric and illumination conditions.

The perception systems we develop are capable of processing previously captured data logs in *playback mode* and recomputing their outputs using software identical to the configuration used on the vehicle during a real autonomous

run. As a result, we can reprocess the same data using different versions of the perception system an unlimited number of times. Current data storage technology makes it easy to record large amounts of raw data at very low cost.

The greatest challenge consists in generating the ground truth information corresponding to the raw data. Our system relies on two methods for generating ground truth information:

1) Surveyed obstacle courses. For many applications, it is possible to construct courses which are highly representative for the type of environment that the robot is expected to perform in. Such courses are typically set up for a particular application, and can be densely populated with the types of obstacles and terrain features a perception system is expected to identify. The data logs recorded while traversing them will contain many example of situations that are interesting for analyzing perception performance. For example, for a farm application an obstacle course might contain different types of fences, poles, human mannequins, ditches of different sizes and other vehicles.

Once an obstacle course is setup, a hand-held RTK GPS unit is used to survey the contours of each obstacle, which are inserted in a database along with information about the nature and size of each obstacle. The idea behind using a surveyed obstacle course is that although a significant amount of effort goes into setting it up, the ground truth information for any data collected over that course will automatically be available. This is possible if the vehicles used for recording data are also equipped with high accuracy positioning systems –even if those position estimates are *not* used by the perception system– such that at any moment in time we know where the vehicle is located on the course and what obstacles it should be detecting based on their GPS surveyed contours.

One of the user interfaces we developed for managing the course survey information and inserting into a database is shown in Figure 2. The advantage of using the surveyed course approach is that massive amounts of data can be collected in a cost-effective manner after the initial investment in the course. This enables us to collect data with a large set of varying parameters such as the environmental conditions, the speed at which the vehicles are driven, the orientation or location of the sensors, etc.

2) Labeling by human experts. In certain cases it is impractical to set up a surveyed obstacle course, or it is necessary to perform an analysis based on data collected in a less constrained fashion in the real application environment. For these situations we have developed the user interface and the procedure described in Figure 3, which imposes two restrictions to vehicle sensor suite:

- It must contain a forward looking camera
- It must contain a range sensing device (stereo camera or laser range finder)[2].

This approach leverages the global positioning system of the vehicle and the precise estimates of the 3-D rigid transforms between the vehicle and the camera and the range sensor in order to obtain the locations of the obstacles in a global GPS referenced frame prior to inserting them in an obstacle database.

[2]unless the user is willing to make certain assumptions about the planarity of the operating environment and the possible shapes of the obstacles

Fig. 3. Obtaining ground truth data from human experts. The user chooses an image offering a good view of an obstacles, and marks the contour of the obstacle. The 3-D points which project inside the obstacle contour in the image plane are marked as obstacles (lower-right). A range filtering tool (upper-right) allows the user to discard points that project inside the obstacle contour but to do physically belong to the object. Using the known position of the vehicle in the GPS global frame one can map the 3-D obstacle points to the global frame as well and insert them into an obstacle database.

While more labor intensive than the previous ground truth acquisition approach, hand labeling is a good solution for medium and short data logs which are collected in situations where time constraints do not allow for a prior survey of the obstacles. As an alternative to exhaustively labeling all data sets or manually selecting the representative scenes, one could apply the techniques described in [13] to larger datasets and only label those scenes that are different from scenes already represented in the obstacle database.

Regardless of the method used for acquiring ground truth data, the end result is that a database is populated with a list of obstacles corresponding to each driven path, on an obstacle course or not. The GPS coordinates of the obstacle contours are stored in the database as well, along with descriptions of the obstacle nature and size.

B. Analysis Support Infrastructure

In order to satisfy the challenging set of the requirements for the analysis component of the PVS, we have decided to build the entire system around a relational database that manages almost all of the data related to the validation experiments: the input data logs, the parameters describing the configuration of the perception system to be tested, the parameters describing which components of the perception system are analyzed, the performance metrics to be applied, their parameters and the results of applying them to the output of the perception system. Our current implementation uses PostgreSQL, a widely used and mature open-source relational database system.

One of the main reasons a relational database engine is the correct choice as the backbone of our validation system is the need for extreme flexibility in the supported queries. Given that the necessary relationships between the different types of data are reflected in the structure of the database, one

can always formulate new types of queries even long after the experiments were performed. Furthermore, we keep track of the outputs produced by the perception system several times per second as opposed to just keeping track of the results for an entire run. This give our system much more flexibility in the type of analyses it can perform, but it results in several tens of thousands of new data entries even for a medium sized set of data log. Modern database systems are specifically designed to support huge amounts of data and queries, which makes them a great fit for our needs.

While the scope of this paper precludes us from describing the inner structure of the database in detail, we will present the main steps involved in going from a set of raw data logs with ground truth information to a set of performance measurements. Unless otherwise specified, all the described functionality is available through a PHP-based web interface to the PostgreSQL server.

(1) **Selection of data logs**

Our data logs contain meta-data about the environmental and illumination conditions, the data collection site, the length in time and distance traveled, etc. Since the log files and the meta-data are managed by the database, the user is able to search for data logs that fit the intended analysis based on any combination of these characteristics and select the ones that will be used. The interface allows the user to upload new data logs to the database, and the meta-data fields can be easily extended.

(2) **Selection of perception outputs to monitor**

In its current form the PVS relies on the fact that our perception systems produce 2-D grid maps of the environment surrounding the vehicle, and they can be compared to the ground truth information. This assumption is in reality not a strong one: most mobile robots have some representation of their surroundings for the purpose of motion planing or obstacle detection; the PVS can be adapted to other representations in order to support systems that do not use 2-D maps.

Our perception systems have several algorithm modules which can produce different types of maps : ground elevation maps, obstacle, tree, slope or ditch maps (see [14], [15]). The PVS enables the user to select which maps will be captured during log playback and hence what data will be available for analysis. For example, one could choose to capture the obstacle map produced by using both laser range finders and cameras and the one produced only by using laser data, so that the performance benefits obtained by adding vision capabilities are measured. It is also possible to select a subset of interest out of the ground truth obstacles which are recorded in the database, to support analyses focused on a particular capability (e.g. detecting ditches).

(3) **Selection of perception software configurations**

Any perception system has some parameters. In our case they are all stored in a directory of text files, which together determine the configuration and behaviors of our perception system. Archived versions of parameter sets are managed by the database, and the user can choose to use one of existing sets, upload a new set of parameters or edit an existing one.

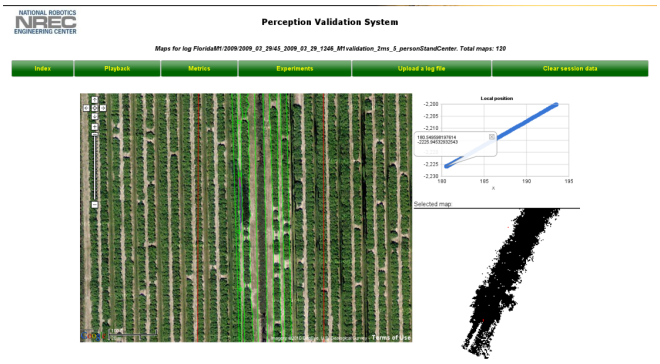


Fig. 4. Web interface for visualizing the output maps captured after the playback stage. The user can see the driven path overlayed on an overhead image and the pairs of output and ground truth maps generated at any point along the path.

As a result, a user with in-depth knowledge of the perception system can have full control over any parameter, while a user with less expertise can use one of the parameter sets already existing in the system.

(4) **Running the perception system in playback mode**

Once the data logs and the parameters to use are selected, the PVS can start playing back all the selected raw logs and feeding then in the perception system. The perception system produces 2-D grid output maps which are then captured by a PVS specific tool along with the time stamp and the vehicle position at the time the map was generated. The information stored in the database.

When the log playback process is completed, the PVS reprocesses all the output maps, and it uses the vehicle position information and the ground truth information from the database in order to generate synthetic ground truth maps which are perfectly aligned with each output map. These ground truth maps are directly linked to the output maps for future queries.

(5) **Inspecting perception outputs manually**

After the playback stage is completed, the PVS user can visualize the output and the ground truth maps using the interface shown in Figure 4. This is a fast way to get a first glimpse at the comparison between the desired and the actual outputs in a georeferenced context.

(6) **Adding, selecting and configuring performance metrics**

A very nice feature of the PVS is that it allows users to independently implement new performance metrics which are best for their particular analysis needs and upload them to the PVS server. The only constraints on the metric "modules" are the following:

- they need to executable from the command line and take two parameters (the output map file and a corresponding ground truth file) as input
- they need to generate their output in XML format.
- the user needs to define the type and names of the input parameters for the metrics and the type and names of the output variables.

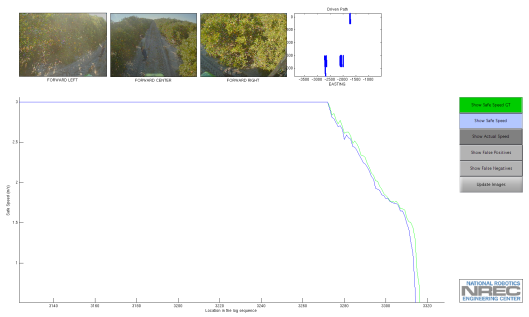
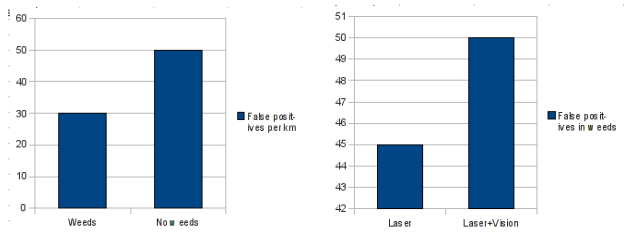


Fig. 5. Innterface for analyzing the results produced by the Safe Speed metrics described in Section III-C. CRIS WILL ADD A LESS UGLY PICTURE OF THIS



(a) (b)

Fig. 6. THIS DATA IS FAKE!

As a results, metrics can be implemented in any programming language. Typical examples would be metrics that counts the number of missed or false obstacle cells, or the *Safe Speed* metric discussed in Section III-C.

(7) **Analyzing perception outputs using performance metrics**

Once one or more performance metrics are selected, the PVS system retrieves all the pairs of output and ground truth maps from the database which apply to the current analysis. The metrics are applied to the map pairs and their outputs are aggregated in the database.

(8) **Analyzing the experiment results**

The PVS provides two ways in which the aggregated performance metric outputs can be analyzed. The first consists in the web interface, which provides simple summaries such as histograms and plots of the outputs of the metrics, along with the possibility to visualize pairs of output and ground truth maps. However, given the huge flexibility offered in terms of the outputs of the performance metrics it is hard to pre-program any in-depth analysis tools. To address this challenge, the PVS supports a second analysis mode: the users can download through the web interface all the outputs of the performance metrics in various formats such as Matlab files, Excel spreadsheets or CSV files. The users can then employ any software of choice to analyze the data. Once such analysis tool is shown in Figure 5.

It is important to notice that since the PVS system is based on a standard PostgreSQL database, it is possible and relatively simple to develop standalone tools independent from our web interface for performing any of the steps above. This has already happened with some of the early testers of our system.

C. Safe Speed: The Performance Metric That Matters

Based on our experience with the PVS, the most useful performance metric we developed is based on the *Safe Speed* estimate, presented in [14]. We will discuss why this type of metric is more useful than comparing detections between the output and the ground truth maps at the map cell level.

The concept of safe speed is simple: given an obstacle map of the area surrounding a vehicle, compute the maximum speed at which the vehicle can move while still being able to stop safely for the obstacles present in the map. The computation is non-trivial, but in essence it takes into account the deceleration capability of the vehicle, its shape, the distance to obstacles and their location relative to the intended path. This makes intuitive sense: there is no need to stop for an obstacle that is very far from the vehicle or outside its path, and it is possible to drive faster if the braking distance is shorter. The safe speed makes the vehicle come to a gentle stop as it nears an obstacle, and it reduces speed when it has to drive close to them.

The main problem with comparing output and ground truth maps cell-to-cell comes from inconsequential detection errors (false positives or false negatives). The performance of most perception systems degrades with distance, and as a result it is often the case that far range obstacle detections are unreliable. However, these errors are also inconsequential: a spurious obstacle detected 50m away from a slow moving vehicle is typically cleared once the vehicle gets closer and it has no effect on any driving decision. Another type of errors with very small consequences relate to possible misalignment of the the obstacles with the ground truth. The contour of an obstacle can be inaccurately recovered (for example due to occlusions, or small errors in the 3-D reconstruction), yet the behaviour of the system will be virtually unchanged; a perception system that locates all obstacles 5% closer than it should will still stop safely in all the necessary situations.

The safe speed metric is based on comparing the estimates generated by performing the computation over the ground truth map (i.e. the map generated if perception was perfect) and over the actual perception produced output map. The metric results can be aggregated in a plot similar to the one shown in Figure 5. Discrepancies between the ground truth and the actual safe speed typically indicate perception errors. We have developed tools which can analyze these plots and extract statistics about the frequencies and the types of errors encountered by the perception system. Two simple examples are shown in the next section.

IV. SAMPLE ANALYSIS RESULTS

This paper is focused on the Perception Validation System and not on any perception system in particular. As a result, we only present results from two simple experiments designed to showcase the types of analyses one might perform.

In Figure 6 we present the results of using a set of data logs captured with an autonomous vehicle in an orchard

environment. For both experiments we use as a metric the number of false positive detections per kilometer traveled as determined using the safe speed metric.

In Figure 6(a) we compare the false positive frequency of the same perception system when operating in orchard rows where tall weeds are present with the performance observed in rows that had been mowed recently and did not have any weeds. This type of analysis could be performed to determine if a given system is capable of functioning in a work environments where weeds are present.

In Figure 6(b) we compare the performance of two different configurations of a perception system over a set of data logs contaning orchard rows with weeds. The two configurations correspond to a system that uses only laser range finder information to detect obstacles, and one that also adds vision based features to the range information. This type of data could be used to analyze trade-offs in autonomous navigation sensor package design.

V. CONCLUSIONS AND FUTURE WORK

We have described a system for measuring the performance of perception systems for autonomous mobile robots. The system is centered around a relational database infrastructure which makes it extremely flexible yet relatively easy to use.

The development of the Perception Validation System addresses two critical needs of the mobile robotics field on the path to commercialization:

- It enables researchers and system developers to make design decisions based on large amounts of data instead of smaller amounts of data and intuition. We believe this will lead to accelerated innovation in our field.
- Perception performance verification is necessary prior to any large scale deployment of autonomous vehicles in the commercial or military domains. This is particularly true in the case of safeguarding system for large vehicles such as cars, construction and farm equipment.

The primary limitation of the PVS system is related to the cost of acquiring ground truth data which is still non-trivial. Either setting up obstacle courses or labeling data by hand requires a significant amount of work, especially if done on a large scale. We are planning to develop additional methods for acquiring ground truth data that would make the process less labor intensive.

An additional limiation comes from the fact that the PVS system is designed to use pre-driven paths. As a result, it is useful for measuring perception performance, but in its current form cannot be used for analyzing the performance of other mobile robot subsystems such as motion planning.

As future work, we intend to continue to acquire data logs for the PVS system and try to investigate and address scalability limits when we encounter them. An additional intention is to package the PVS as standard set of tools in order to make it applicable to any other perception system that adheres to a minimal interface. We believe that this capability would have a significant positive impact on the repeatability and quality of mobile perception research.

VI. ACKNOWLEDGMENTS

The authors gratefully acknowledge the support received from Jose Gonzalez Mora for adapting the hand-labeled ground truth user interface to the needs of the perception validation system. This work was supported by Deere & Co. under contract XXXXXX.

REFERENCES

[1] E. Krotkov, *et al.*, "The DARPA PerceptOR evaluation experiments," *Autonomous Robots*, vol. 22, no. 1, pp. 19–35, Aug. 2006. [Online]. Available: http://www.springerlink.com/index/10.1007/s10514-006-9000-0

[2] N. F. Drive, E. Krotkov, M. Perschbacher, and J. Pippine, "The DARPA LAGR Program : Goals , Challenges , Methodology , and Phase I Results," *Journal of Field Robotics*, vol. 23, no. 2006, pp. 945–973, 2007.

[3] L. Jackel, D. Hackett, E. Krotkov, and M, "How DARPA structures its robotics programs to improve locomotion and navigation," *of the ACM*, vol. 50, no. 11, pp. 55–59, 2007. [Online]. Available: http://portal.acm.org/citation.cfm?id=1297823

[4] A. Lakhotia and E. P. Blasch, "Come of Age :," *Computer*, pp. 6–9, 2006.

[5] C. Urmson, *et al.*, "High Speed Navigation of Unrehearsed Terrain : Red Team Technology for Grand Challenge 2004," *Intelligence*, 2004.

[6] S. Thrun, *et al.*, "Stanley: The robot that won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, Sept. 2006. [Online]. Available: http://doi.wiley.com/10.1002/rob.20147

[7] M. Montemerlo, *et al.*, "Winning the DARPA Grand Challenge with an AI Robot ," *Artificial Intelligence*, no. Gat 1998, pp. 982–987, 2006.

[8] C. Urmson, *et al.*, "A Robust Approach to High-Speed Navigation for Unrehearsed Desert Terrain," *Journal of Field Robotics*, vol. 23, no. 8, pp. 467–508, 2006.

[9] ——, "Autonomous Driving in Urban Environments : Boss and the Urban Challenge," *Journal of Field Robotics*, vol. 25, no. February, pp. 425–466, 2008.

[10] J. Bagnell, *et al.*, "Learning for Autonomous Navigation," *Robotics & Automation Magazine, IEEE*, vol. 17, no. 2, pp. 74–84, 2010. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5481587

[11] T. Fawcett, "ROC graphs: Notes and practical considerations for researchers," *Machine Learning*, vol. 31, pp. 1–38, 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.10.9777&rep=rep1&amp

[12] J. Vanschoren, "Understanding Machine Learning Performance with Experiment Databases," *lirias.kuleuven.be*, no. May, 2010. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Understanding+Mach

[13] C. Dima and M. Hebert, "Active learning for outdoor obstacle detection," in *Proceedings of the Robotics Science and Systems Conference, Cambridge, MA*. Citeseer, 2005. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.8867&rep=rep1&an

[14] S. Moorehead, C. Wellington, H. Paulino, and JF, "R-Gator: an unmanned utility vehicle," *Proceedings of SPIE*, 2010. [Online]. Available: http://link.aip.org/link/?PSISDG/7692/769215/1

[15] D. Johnson, D. Naffin, J. Puhalla, and J, "Development and implementation of a team of robotic tractors for autonomous peat moss harvesting," *of Field Robotics*, no. Figure 1, 2009. [Online]. Available: http://www3.interscience.wiley.com/journal/122262233/abstract