Data Collection Test-Bed for the Evaluation of Range Imaging Sensors for ANSI/ITSDF B56.5 Safety Standard for Guided Industrial Vehicles

Will Shackleford
National Institute of Standards and Technology(NIST)
100 Bureau Drive, Stop 8230
Gaithersburg, MD 20899
(301) 975-4286
shackle@nist.gov

Roger Bostelman
National Institute of Standards and Technology(NIST)
100 Bureau Drive, Stop 8230
Gaithersburg, MD 20899
(301) 975-3426
roger.bostelman@nist.gov

ABSTRACT
In this paper, we describe the process by which we collected sensor data for the evaluation of 3D LIDAR (Light Detection and Ranging). Data were also collected simultaneously from SONAR (Sound Navigation and Ranging) sensors, navigation systems, 2D Laser Measurement Sensor (LMS) and a color camera. We describe software developed to perform data collection and allow for evaluation of the data both offline and in real-time during the data collection and briefly cover the experiments themselves where various obstacles were placed in front of a moving vehicle and results were recorded as to whether the obstacle was detected or not.

Categories and Subject Descriptors
I2.10 [**Vision and Scene Understanding**]: 3D/stereo scene analysis

General Terms
Performance, Design, Experimentation, Standardization,

Keywords
LIDAR, LADAR, Data-Collection, Autonomous Guided Vehicles(AGV), B56.5, Sonar.

1. INTRODUCTION
The Industrial Truck Standards Development Foundation (ITSDF) manages the "ANSI/ITSDF B56.5 Safety Standard for Guided Industrial Vehicles and Automated Functions Of Manned Industrial Vehicles" as approved by the American National Standards Institute (ANSI)[2]. The National Institute of Standards and Technology (NIST) has been performing measurements to be used as background information towards changes in the standard. Automated Guided Vehicles (AGVs) are typically programmed to follow prescribed paths but still need sensors to detect obstacles such as closed doors, equipment, personnel or material left temporarily in the vehicle's path. Currently they rely heavily on 2D line scanners with a physical bumper as the final backup to stop the vehicle in these cases. The 2D line scanners work well against most vertical obstacles but it takes many of them to completely protect against overhanging obstacles and even then they do not scan the full volume of space the vehicle will travel through. Flash LIDAR is a relatively new class of range imaging sensors with the potential to scan 3D volumes faster than the line scanning systems. To evaluate them, a consortium of AGV vendors was formed that took preliminary data with several Flash range imaging systems and selected one for further development and investigation. This is the sensor used for this work. In 2008 work was done with the stationary vehicle and with a manually moved cart.[1] In 2009, the data collection system was integrated with the Mobility Open Architecture Simulation and Tools (MOAST) framework.[3] This allowed the system to collect data while being driven autonomously.

PerMIS'09, September 21-23, 2009, Gaithersburg, MD, USA.
ACM 978-1-60558-747-9/09/09

* Certain commercial equipment, instruments, or materials are identified in this paper to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

2. Test-Bed Hardware
All of the sensors are mounted on a robot. The sensors include:*

- Spinning Laser Positioning System (SLPS) - Provides absolute position using a spinning laser that detects special reflective targets mounted on walls and fixed structures.
- Safety Laser Measurement Sensor(LMS) – 2D line scanner which detects obstacles but only at a single height.
- FLASH – 3D Flash LIDAR Camera provides range/intensity for every pixel in the image.
- Color Camera – Provides better documentation of each test.
- Positioning Camera (CamPos). -- A camera system pointed at the ceiling to provide absolute position using special targets mounted on the ceiling.

- SONAR – Sound navigation and ranging sensors mounted to the vehicle to detect obstacles in the vehicle's path.

The positions of the sensors are shown in Figure 1.

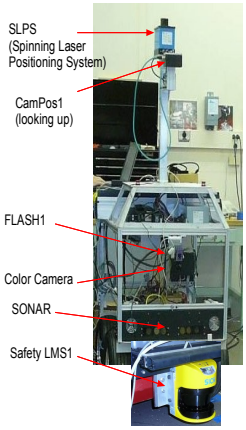

Figure 1: Sensor Positions

3. Software Architecture

3.1 Main Software Architecture

Figure 3.1 provides the main software architecture diagram for the system. The moastLogRecordSuper supervises and coordinates MOAST controller moves with the starting and stopping of the data collection. The MOAST framework aids in the development of autonomous robots. It includes an architecture, control modules, interface specs, and data sets and is fully integrated with the USARSim (Unified System for Automation and Robot Simulation) system.[3] The MOAST controller is used to generate trajectories for Player and send a stream of desired translation and rotational velocities to Player. Player is a cross-platform robot device interface and server that supports a number of robot platforms and sensors including the commercial platform used as our base and LMS1.[5]

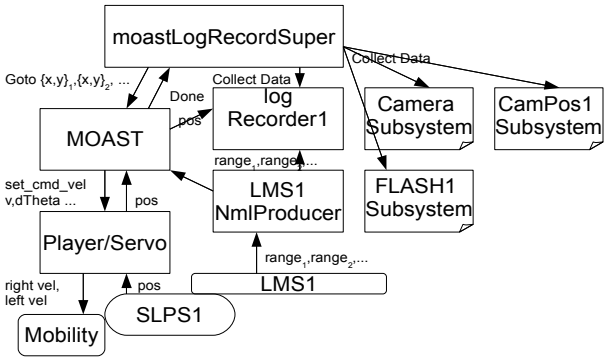

Figure 2: Main Software Architecture Diagram

3.2 Neutral Message Language(NML)

The Neutral Message Language (NML) [4] provides both the communication system and the facilities for reading and writing the data files in a portable, transport, platform and programming language independent manner. NML is part of the Real-time Control System (RCS) Library[11]. It provides a common API (Application Programming Interface) for both potentially remote TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) communications as well as faster shared memory based communications. It is used for all inter-process communications within the system including internal MOAST communications with the exception of the communication between MOAST and Player [5] which uses a Player-defined socket interface.

3.3 Sensor Subsystem Software Architecture

Each sensor is handled by a similar subsystem as shown in Figure 3.2 Each sensor comes on a data bus such as Firewire, USB, RS232 Serial, or Ethernet with a format and protocol usually unique to that model sensor. A separate process is used for each sensor that essentially acts as a device-driver and converts data received from the bus to an NML message and writes it to both a queued and non-queued NML buffer. The queued buffer is read by the LogRecorder which then writes the data to disk. A non-queued buffer is used to provide a real-time display of the data. The real-time display is needed during data collection to ensure that the sensor is working, and that objects of interest are within the field of view and to adjust sensor configuration parameters. Using separate processes for the NmlProducer and LogRecorder means that intermittent delays in writing to the disk will not cause the system to miss frames as the NmlProducer could continue filling the QueuedBuffer while the LogRecorder is delayed. It also isolates the code most likely to need debugging and not to be portable, which is the sensor specific driver code. Each viewer is built so that it can display either live data from the NML channel or logged data from files.

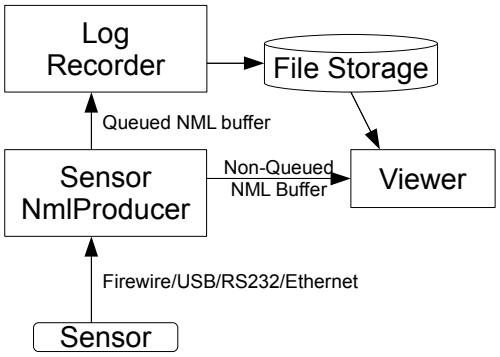

Figure 3: Sensor Subsystem Architecture

The LogRecorder and Sensor NmlProducer are written in C++ for performance and low-level access to hardware or operating system resources. Each viewer is written in Java so that the logged data may be evaluated on any platform.

3.4 MOAST

MOAST is a control system framework that works both in simulation and on real hardware. The simulation uses a commercial gaming engine and is developed under the Performance Simulation Project at NIST.[3][10] Figure 4 shows the USARSim module/plugin showing a small simulated robot in a warehouse setting and the RCS Diagnostics tool connected to the bottom 3 levels of the MOAST (AM,PRIM,SERVO). The Autonomous Mobility (AM) level combines data from the various sensors into a map and uses the map to compute a path from the vehicle's current position to the final goal position that avoids all known obstacles. The PRIM level takes a list of intermediate positions or arc segments produced by the AM level and considers the dynamics and kinematics of the vehicle to produce a series of SERVO commands sent one at a time as the vehicle moves along the path. The SERVO level takes commanded right and left wheel velocities and interfaces with the hardware to achieve those velocities. The moastLogRecordSuper can connect either to the PRIM or AM levels. When commands are sent directly to PRIM the robot will simply follow a set of way-points and ignore the sensors. When commands are sent to the AM level the robot will use the sensors to build a map and plan around obstacles. All of the tests done so far have sent commands directly to the PRIM level which gives us greater control over exactly where the robot travels but the option of using the AM level for future tests remains available.

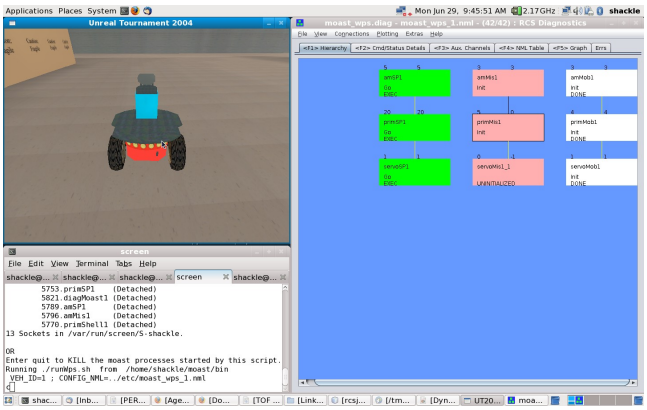

Figure 4: Left: Unreal Tournament with USARSim module simulation/animation, RIGHT: RCS Diagnostics tools showing AM,PRIM,SERVO levels of MOAST.

3.5 NML Packed Message Files

All of the data after collection is stored in NML packed message files. The format will hopefully provide the openness and flexibility of text, CSV (comma-separated values)[8] or XML (Extensible Markup Language)[9] files and the efficiency in both disk space and processing time of binary formats. This file format allows for easy reading and writing of even complex data structures. Generic tools can be used to display the contents of the files or users can write their own programs to read and write the files with a simple API. Also a memory map file listing the offset to every variable allows the files to be accessed outside the API. Tools were written to convert these files to plots, movies, and separate still image files as appropriate.[6] A website is under development that should allow users to access the collected data using any of the tools discussed below, download the message files, export appropriate subsections to a spreadsheet , and etc.

The file sizes for one LIDAR data frame for the FLASH1, which has a resolution of 144 pixels x176 pixels (25344 pixels), are compared in Table 1. The packed format includes configuration information and XYZ coordinates from every pixel as well as a range and intensity value. Obviously if a display is all that is needed, saving JPEGs requires by far the least data but it does not include configuration data or the XYZ point cloud and even the exact range and intensity values cannot be recovered from the JPEG. The CSV and spreadsheet files with the same data are larger and do not contain the configuration information or XYZ coordinates although those could be added. There are many ways the information could be stored in XML, but one of the most straight-forward methods produced a file six times larger than the original packed data.

File Format	XYZ ?	Config.?	Size(Bytes)
XML	Yes	Yes	3047494
Spreadsheet	Yes	No	2829824
CSV	Yes	No	1374147
NML Packed	Yes	Yes	507007
JPEG	No	No	3382

Table 1: LIDAR Data Frame File Sizes

3.6 Flash LIDAR Display

Flash LIDAR cameras generally provide both a range image (Figure 5) and and intensity image (Figure 6). Although all tests done this year were with the FLASH1, the data structure used to store the data was originally developed with two other Flash Lidar Cameras and therefore all tools should work with all three cameras. It is often easier for people to see objects in the intensity image. However, it is the range image that is of most use for the AGV's obstacle detection and avoidance algorithms. One goal of the experiments was to determine whether the obstacles were detected by the sensor at various ranges. Unfortunately this could be somewhat subjective. Just because a person (especially one already familiar with the scene) could make out something is no guarantee that it is possible to use the data to build a reliable automatic obstacle detection algorithm. For this reason the viewer includes its own obstacle image classification window. The results of the obstacle detection are shown in Figure 7.

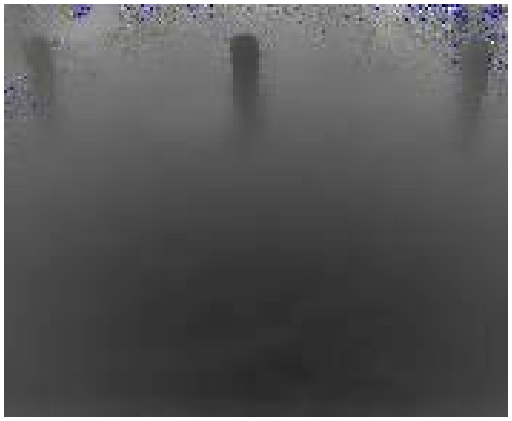

Figure 5: Flash LIDAR Range Image of 3 Cylinders in front of AGV (Black=near, White=far)

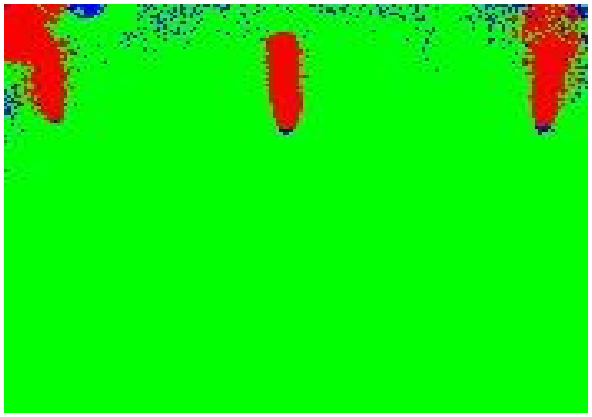

Figure 7: Obstacle Detection Based On Flash LIDAR Range Image (Green=ground,Red=Obstacle, Blue=Unknown)

Figure 6: Flash LIDAR Intensity Image of 3 Cylinders in front of the AGV

The obstacle detection includes filters to eliminate points with too high/low intensity, too high/low range values, isolated points or points not near neighboring points. It then simply rotates the point cloud to adjust for the mounting of the sensor and applies a height threshold. Points below the threshold are ground and points above the threshold are obstacles. All the parameters are adjustable both in real-time and when displaying logged data from a Graphical User Interface to allow for a very conservative to very lenient obstacle detector. A text display allows min/max and average intensity or range values to be obtained for any selected rectangle in the image.

The sensor has two problems. First, it cannot distinguish obstacles at multiples of its modulation wavelength (about 6 m). i.e., if the modulation wavelength was 6 m an object 7 m away returns the same value as one 1 m away. Second, when near highly reflective surfaces such as the ones commonly used by the AGV's navigation system the entire scene is strongly distorted. For this reason all of our tests have the sensor pointing down towards the floor which eliminates any possibility of an object being farther away than the modulation wavelength and also keeps the sensor away from the eye-level navigation reflectors.

3.7 Camera Positioning Navigation System/Spinning Laser Positioning System

Spinning Laser Positioning System (SLPS) is typically used for industrial AGV's. CamPos is a more recent alternative to the spinning laser based navigation systems that uses a camera and 2D bar code targets mounted on the ceiling.[7] One set of tests that we completed was to record both the CamPos and SLPS positions simultaneously while driving the ATRV manually. We purposely mounted the 2D bar code targets in a fairly regular ceiling pattern of 1.2 m spacing and disregarded partially occluded ceiling obstructions. Where obstructions mostly or completely covered targets, we moved those targets to a less obstructed ceiling location. Although the manufacturer suggests non-obstructed targets, we are looking for ways to measure performance of these systems when they are in the ideal and non-ideal configurations. Figure 8 (left) shows clear view and partially occluded views of ceiling-mounted 2D bar code targets. A similar situation can occur with the SLPS positioning of wall-

mounted reflectors as shown in Figure 8 (right). Here, reflectors are shown in clear view and partially occluded views that could also result in less than robust vehicle positioning. In previous tests, we also found issues with this system when highly reflective surfaces appear to the sensor as system reflectors.

For this recent experiment, we tested only the CamPos system targets being partially obstructed. The experiment showed that while the CamPos tracked the SLPS position well over much of the approximately 36 m long x 10 m wide course, there were measurement issues in places where the CamPos could not simultaneously see more than one target (see Figure 9 left and right). These were caused by an overhead crane system (as shown in Figure 8 (left)) and ceiling supports that obscured a few bar code targets.

The user of both the CamPos and SLPS systems can ideally mount sensor targets appropriately to get maximum accuracy as specified by the manufacturer. As targets are relatively inexpensive for these sensor systems, adding more and calibrating the targets mounted in non-occluded areas easily solves these issues.

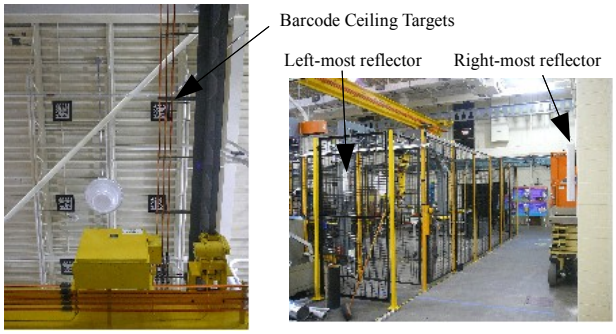

Figure 8: (left) Crane electrical bars partially occluding the ceiling-mounted 2D bar code targets of the CamPos system; (right): clear view of the right-most reflector of the SLPS system and partially occluded view of the left-most reflector by a robot cage

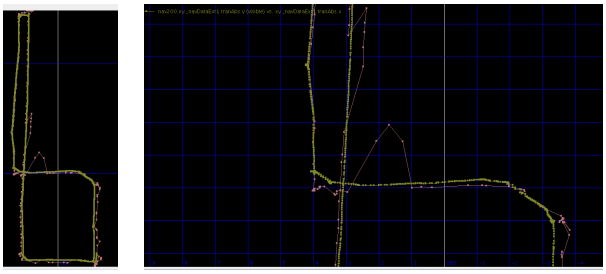

Figure 9: (left): CamPos (pink) versus SLPS position (green/yellow) plots; (right): Zoom of CamPos versus SLPS position where ceiling barcodes are partially occluded.

Both the CamPos and SLPS data are recorded with independent running implementations of the sensor subsystem as described in section 3.3. The data was plotted from the recorded data offline using the plotter included with the RCS(Real-Time Control System) Diagnostics Tool.

3.8 Safety Laser Measurement Sensor (LMS)

The LMS is a laser scanning system that currently detects obstacles at longer ranges and higher reliability than the Flash LIDAR but only in a single plane. In Figure 10 the LMS was scanning through a fence that will be placed around a robot work station. In the raw sensor data the LMS sees both the fence and the object on the other side of the fence.

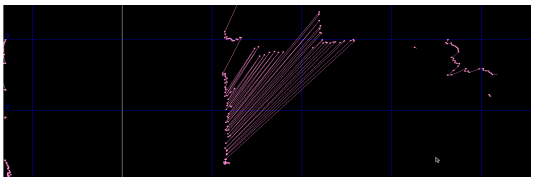

Figure 10: LMS1 data scanning both a fence and obstacles on the other side of the fence.

3.9 Color Camera

The main use of the color camera currently is to overlay images from the 3D Flash LIDAR to better identify the source of artifacts as shown in Figure 11.

Figure 11: Color camera image overlaid with obstacle detection data from the flash LIDAR sensor

4. Static Test Results

A series of static tests were performed using the sensors listed in section 2 Test-bed Hardware. The tests included covering three different test pieces with a variety of surfaces and testing with all sensors recording at a variety of positions, orientations and ranges. The test pieces included the two pieces already part of the B56.5 standard, a 200 mm diameter x 600 mm long horizontal cylinder and a 70 mm diameter x 400 mm high vertical cylinder. The third test piece was the new 500 mm x 500 mm flat surface target. The coverings were selected to change the reflectivity and specularity of the test pieces for the optical sensors as well as the sound absorption properties for the SONAR.

For both optical and SONAR sensors, changing the angle of reflection with the flat target had a significant effect on the measured intensities and ranges and in some cases whether the

sensor received a return or not. The reflectance and specularity of the coverings also had an effect on the optical sensors however none of the sound absorbing materials tested had a significant effect on the SONAR.

5. Changes to the ANSI/ITSDF B56.5 Safety Standard for Guided Industrial Vehicles

As a result of tasks conducted using this test bed, changes were recommended to the ANSI/ITSDF B56.5 committee to add an additional flat target, to test at a variety of reflectance and specularity values and at different reflection angles and ranges. There was also a recommendation to perform dynamic tests at various vehicle speeds. We may provide further recommendations after completing the dynamic tests.

5.1 Conclusions

NIST has created a unique test bed and data-collection platform. Although its initial use was to provide input to the ITSDF standard development process, it should be possible to provide industry and/or the research community with independent evaluations of sensor technologies or provide data for obstacle detection algorithm development or verification. The test bed will continue to be updated with additional sensors.

6. REFERENCES

[1] Roger Bostelman, MS; William Shackleford, "Time of Flight Sensors Experiments Towards Vehicle Safety Standard Advancements", submitted to Computer Vision and Image Understanding Journal

[2] Industrial Truck Standards Development Foundation, (2005). ITSDF B56.5 Safety Standard for Guided Industrial Vehicles and Automated Functions of Manned Industrial Vehicles, http://www.itsdf.org.

[3] Mobility Open Architecture Simulation and Tools (MOAST) framework, https://sourceforge.net/projects/moast

[4] Neutral Message Language (NML) , http://www.isd.mel.nist.gov/projects/rcslib/NMLcpp.html

[5] Player Project, http://playerstage.sourceforge.net/index.php?src=player

[6] NML Message Files, http://www.isd.mel.nist.gov/projects/rcslib/message_files.html

[7] SkyTrax System, http://www.sky-trax.com/products/STS.php

[8] Wikipedia: Comma-separated values, http://en.wikipedia.org/wiki/Comma-separated_values

[9] W3C Extensible Markup Language(XML), http://www.w3.org/XML/

[10] Performance Simulation Project, http://www.nist.gov/mel/isd/ks/persim.cfm

[11] Real-Time Control Systems Library, http://www.isd.mel.nist.gov/projects/rcslib/