Robot Simulation Physics Validation

C. Pepper, S. Balakirsky, and C. Scrapper
Intelligent Systems Division, National Institute of Standards and Technology (NIST)
Gaithersburg, MD 20899-8230, USA
Email: {Christopher.Pepper, Stephen.Balakirsky, Christopher.Scrapper} @NIST.Gov

***Abstract* —** Computer simulation of robot performance is an essential tool for the development of robot software. In order for simulation results to be valid for implementation on real hardware, the accuracy of the simulation model must be verified. If developers use a robot model that is not similar enough to the actual robot, then their results can be meaningless. To ensure the validity of the robot models, NIST proposes standardized test methods that can be easily replicated in both computer simulation and physical form. The actual robot can be tested, and the computer model can be finely tuned to replicate similar performances on equivalent tests. To illustrate this, we have accomplished this task with the Talon Robot[1] on NIST standard test methods..

***Keywords*:** *performance metrics, physics validation, response robots, robots, simulation, urban search and rescue, USARSim*

I. INTRODUCTION

There is a growing trend in intelligent systems research to use a simulated environment in the initial phases of development. As simulations become more integral in the development process, it is important for them to become more accurate to protect the validity of experiments. The solution to this is to develop standard test methods [1] and validate the performance of the robot on the test methods in both reality and simulation [2].

A. The Benefits of Computer Simulation

There are a myriad of benefits to computer simulation for a researcher that make it an attractive option during the development process. An important attribute of simulations to a developer is repeatability, which allows for simplified debugging because the same scenario can be precisely generated to trigger a known error and check the solution. In addition to this, all vital data can be logged, including ground truth, to give developers an understanding of inconsistencies in their algorithm performance. In contrast to an actual environment, simulation gives developers access to cost prohibitive or unavailable sensors. Time can also be spent efficiently since many researchers can work on copies of a virtual platform simultaneously where physical platforms may be limited in availability. Additionally, the actual testing environment may not be accessible, or may only be accessible at certain times while the simulated environment is always available. Virtual access to different testing environments makes virtual testing very cost efficient. Simulation is also safer for researchers; and allows them to safely refine their assumptions about the robot and their algorithms. Therefore, computer simulations allow a development team to be more effective and efficient.

B. The Need for Standard Performance Metrics

In some cases, there are errors in the robot models that result in physics inaccuracies with friction, gravity, mass, force, etc. The consequence of this is that the simulation results can be unreliable. In some cases, models exhibit behaviors that are not possible for the actual robots they represent. Researchers cannot accurately evaluate the performance of the robot with a faulty model. The challenge is therefore to develop a method to expose and resolve the inconsistencies between virtual models and real robotic systems.

C. The Proposed Solution

Developers can use standardized test methods [2] to ensure that the model they use behaves as close to the actual robot as possible. Using the test methods reveals unknown inconsistencies between simulation and reality, and researchers can then identify the problem with the physics of the model. One can resolve the issue systematically with an understanding of the simulation physics parameters. These standard test methods can also be used to verify existing model performance. With the virtual models validated, researchers can develop their software and confidently integrate their work onto physical systems.

II. BACKGROUND

A. USARSim Simulation Environment

USARSim is a high-fidelity simulation of urban search and rescue (USAR) robots and environments, and is intended as a research tool. It builds upon a commercially available game engine produced by Epic Games [3] known as the Unreal Engine 2.0. Today's games often achieve a high level of complexity and realism, and the game engines have become general purpose simulation engines that can be used to implement multiple games based on the same foundation.

1 Certain commercial equipment, instruments, or materials (or suppliers, or software...) are identified in this paper to foster understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

They are extremely customizable, and therefore are excellent candidates to be used to develop robot simulators and perform scientific investigations [4].

While the internal structure of the Unreal Engine is proprietary, developers can purchase the engine code. For most uses this has been made unnecessary by the University of Southern California's Information Sciences Institute interface known as Gamebots [4] [5] that allows an external application to exchange bi-directional information with the engine. This interface was created for research in artificial intelligence and is an open source project [5]. The Unreal Engine implements a Virtual Machine, a concept very similar to the Java Virtual Machine, which allows for external code to be executed by the engine. The code must be written in the UnReal host language called UnrealScript, which is an object oriented language with syntax resembling C++ and JavaScript. The code may then be compiled into an intermediate platform-independent bytecode that is executed by the Unreal Engine. Through UnrealScript, a developer has full access to all environmental variables and full control of the actors in the world.

USARSim sits on top of Gamebots and provides a standardized interface to robot actuators and sensors. Extensive research of USARSim, in various applications, has shown the simulation to behave in a predictable manner with high correspondence to reality. This research is detailed in [6], [7], [8], and [9]. USARSim has experienced wide community acceptance with over 17,000 component downloads to date. In addition, it is the basis for the RoboCup Rescue Simulation League Virtual Robots Competition [10]. Additional information on USARSim and related software may be found in [11].

Fig. 1. USARSim Talon Model[2]

B. Karma Parameters

KActors are a class of objects that are controlled by the Karma Physics Engine. Karma is the game engine used by Unreal Tournament to control the vehicle physics, level physics, and rag doll physics [12]. Complicated systems, such as robot manipulators, can be created using Karma joints. Most objects in the simulation are static during game play, like static mesh actors[3]. KActors are dynamic and interactive, and each KActor has general Karma parameters, referred to as KParams, which define its own behavior in the simulation. The KParams that we use in this paper are KFriction, KangularDamping, and KCOMOffset. KFriction ranges between zero and one, where the KActor experiences no friction at a value of zero and total friction at a value of one [14]. KAngularDamping is the parameter that determines the magnitude of force to decrease the angular velocity of the KActor. KCOMOffset is a vector that defines the displacement of the center of mass from the center of the KActor. These are the Karma parameters that dictate most of the actions that we will change in the robot. More information on the Karma Physics Engine may be found in [12].

C. Talon Robots

Talon robots are robots produced by Foster-Miller, Inc. that are used for "explosive ordnance disposal (EOD), reconnaissance, communications, hazmat, security, defense rescue" [15]. We chose the Talon for this paper because NIST has access to the robot, allowing us to determine its capabilities through physical experimentation. It should be noted that the NIST robot is several years old and that newer, more capable Talon models exist.

Fig. 2. Foster-Miller Talon Robot[4] [15]

III. NIST STANDARD TEST METHODS[5]

NIST engineers have developed standard test methods designed to analyze the performance of USAR robots in a repeatable and objective manner [1]. Each test was designed to test a specific attribute of a robot that is determinative of how successful it can be in a range of rescue situations. The

2 Simulation results for particular payload shown in figure 1.

3 Those unfamiliar with static mesh actors should read [13].

4 This picture depicts a robot configuration different than that used by NIST in testing.

5 Additional information about the NIST Reference Test Arenas for Autonomous Mobile Robots can be found in [16].

test methods are being developed in partnership with first responders, robot developers, and technical experts. The following test methods are a few of those created by NIST and others, several of which have been submitted to and approved by the Operational Equipment Subcommittee of the ASTM International E54.08 Homeland Security Committee [17].

A. Directed Perception

The directed perception test is designed to analyze the use of "robotic manipulators to perform a variety of tasks in complex environments" [17]. The test artifacts consist of cardboard boxes of uniform size with cutout holes. Each box has targets inside that different sensors can identify, such as lights and hazmat signs. Non-flat flooring also increases the difficulty of this test.

Fig. 3. Directed Perception Test at 2007 Metro Tech Event, NIST with teleMAX Bomb Disposal Robot

B. Grasping Dexterity

This test method analyzes the "requirement to retrieve objects, not necessarily configured for robot manipulators, within complex environments" [17]. The setup contains stacked shelves with items for the robot to pick up and place from one location to another on the shelving. The items are often blocks, simulated pipe bombs, or water bottles. The flooring is also often variable in terrain.

Fig. 4. Grasping Dexterity Test at 2006 RoboCamp Rome, Italy with teleMAX Bomb Disposal Robot

C. Stairs

Stairs test the ground mobility of a robot. The robots must be able to climb any variety of stairs, including stairs enclosed on the sides, with railings on the sides, with risers, or open stairs [17]. They can be constructed of different materials and at different slopes, presenting a difficult mobility task.

D. Step Field Pallet

The step field pallets are "repeatable surface topologies with different levels of 'aggressiveness'" [17]. A half step field pallet (also known as orange step fields) is classified as medium difficulty mobility, and a full step field pallet (also known as red step fields) is classified as high difficulty mobility. The computer generated random step field pallets are an abstracted test of the mobility of a robot. They are easily recreated and easily reconfigured. The step field pallets simulate uneven ground such as that seen in a rubble pile.

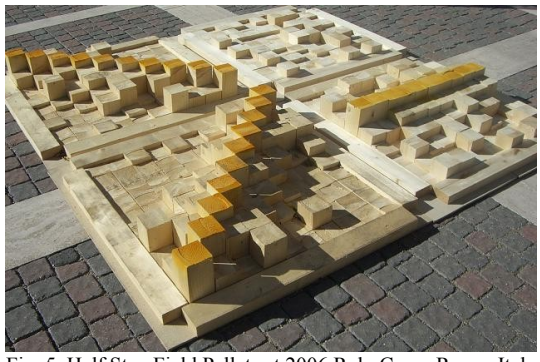

Fig. 5. Half Step Field Pallets at 2006 RoboCamp Rome, Italy

Fig. 6. NIST 30cm Step Test with Pipes

Fig. 7. NIST 20cm Virtual Step Test with Pipes

E. Step Test

The step test is designed to analyze the capability of a robot to climb increasingly higher plateaus. In some challenges, shelving brackets are used to hold polymer of vinyl chloride (PVC) piping at the edge, forcing the robot to not grip onto an edge for leverage. The free-spinning piping also simulates a slippery surface the robot may need to climb.

F. Mobility and Endurance (Zigzag and Figure 8)

These test methods are based on the step field pallet test. The formation of the step field pallets is designed to test the mobility and endurance of the robot. In this task, robots are to traverse a prescribed course of either a figure 8 shape or zigzag shape. Robots must be able to travel the length of the course quickly enough to avoid losing all battery life, and any field-repairs of the robot are timed. In the figure 8, multiple laps may be required.

Fig. 8. Zigzag Endurance Test

Fig. 9. Medium Difficulty Figure 8 Test
with teleMAX Bomb Disposal Robot

IV. VALIDATING TEST METHODS: THE STEP TEST

Prior to using test methods in simulation, we must first create the test methods and ensure *they* perform as expected. Researchers can determine the value of individual physics parameters with simple experiments and reasoned approximation. The model of the step test was created to the exact dimensions of the actual test. The important physics parameters in the real and simulated tests are the friction of the oriented strand board (OSB), the friction of the PVC piping, and the angular damping of the PVC piping.

A. Deriving the OSB Friction of the Step Test

A simple experiment was created to determine the actual frictional behavior of OSB. The test consisted of timing various sizes of OSB sliding on a larger OSB sheet at five different angles. Several trials were performed for each angle. At an angle of 9.9º, the approximately 35.6cm x 35.6cm (14” x 14”) board had enough static friction to resist motion when at rest and enough kinetic friction to slow to a stop quickly when in motion. At 14.8º, the board took approximately 1.9s to slide down the approximately 122cm x 122cm (4' x 4') sheet. This behavior was replicated in a simulation through a heuristic derivation of the KFriction parameter, the final value of which was 0.56. Further testing revealed that this value at the remaining angles produced results with a strong correspondence to reality. These results are shown in Table 1.

TABLE 1

14”x14” PLYWOOD FRICTION TEST RESULTS

	Time for Sheet to Slide Down, seconds							
	Trial Number							
Ramp Angle	1	2	3	4	5	6	7	AVG
14.8°	1.91	/	/	/	/	/	/	1.91
15.2°	1.48	1.64	2.43	1.77	1.66	/	/	1.80
18.5°	1.06	1.27	1.19	1.11	1.06	1.45	1.61	1.25
27.8°	.81	.73	.72	.70	.72	.70	.72	.73

In the derivation of this parameter, several interesting results with the physics engine and its friction were recorded. The first observation was that the KMass of the object had no effect on the friction of the object. One would expect that this parameter was the coefficient of kinetic friction, μ_k, and that it would follow the classic relationship,

$$F=\mu_k N, \quad (1)$$

where N is the normal force and F is the force of the drag, but this was not the case. The second observation was that static meshes, with added KParams or without, do not affect the actions of a KActor. KActors seem to only be affected by other KActors. The pallets used in the step test were changed from static meshes to KActors, to allow the test to affect the vehicles. Because KActors are movable during simulation, the translational motion of them must be controlled. The motion of the pallets was limited by ball and socket joints, a KBSJoint Karma constraint in UnrealEd. These constraints prevent the pallets from sliding out from under the robot during the test.

B. PVC Piping Physics Parameters

Analysis of film from previous USAR events with the step test showed that there was little to no slip between robot tracks and the PVC piping. KFriction provides full friction, i.e. no slip, when set to a value of 1.0. A value of 0.9 for KFriction will allow tracks to mostly grip the pipe with a small amount of slip. Finally, the value of the angular damping parameter needs to be determined because the pipe experiences friction from the shelving bracket in reality. The value of 1.0 was chosen to prevent the pipe from spinning endlessly and to allow the robot tracks to easily spin the pipe.

V. IDENTIFYING MODEL INCONSISTENCIES

Based on behavior analysis of the model, the step test in reality and virtual simulation are now consistent. Testing with these methods may uncover differences between the actual robot and the virtual model. To do this, we simply analyzed data captured on the actual robot as it attempted different tasks on the test. It is important to note that the difficulty of the tests must be increased, for example raising the height of the step, until the physical robot is unable to accomplish the test. This provides an upper bound for what the simulation should be capable of performing. After analysis of this information, we tested the virtual model to determine whether the simulation behavior was accurate. Rigorous comparison shows how the virtual model needs to be altered.

This was the process for analyzing the Talon robot and model. The first experiment on the step test was driving the robot in a direct forward approach, the second was at an angled forward approach (figure 10), the third was a reverse approach (figure 11), and the final experiment was at an angled reverse approach[6]. The same procedure was then repeated for the model in simulation. The results of these tests are shown in Table 2, where a "yes" is climbing the step and a "no" is not doing so. Other observations were recorded, such as issues the implemetation of the approximation of the track behavior.

TABLE 2

RESULTS OF 20CM STEP TEST WITH PIPE

	Robot	Model	Correlates
Direct Forward	No	No	☑
Angled Forward	No	Yes	☒
Direct Reverse	No	Yes	☒
Angled Reverse	No	Yes	☒

6 All of the real and simulated tests were performed with the manipulator arm folded on top of the robot to keep a constant center of gravity. This position is the start pose of the robot arm and can be seen in figure 11.

A. Track Implementation

In the current version of the Unreal Engine, version 2, tracks on vehicles must be approximated. These tracks are approximated in one of two ways. The first has a static tread attached to the robot, and the robot uses the gears (that normally propel the track in reality) to propel the vehicle by directly interacting with the world as wheels. A vehicle model that does this is the teleMAX robot, developed by telerob [18]. Another method used to estimate the behavior of the track is to have many wheels of different sizes approximate the shape of the track.

The second method was used for the Talon, where the Talon has large front and rear wheels and little wheels in between. The small wheels can move translationally to simulate the flexing of the track. Currently, these wheels are rigid and oppose transltational motion. Testing has shown that the wheels on a side, which are supposed to behave as a single track, can spin at different speeds or in different directions, which is not possible for a track. The individual gears all contribute to the motion of the track, which is at a uniform speed at all points on the track. The implementation of the tracks needs correction to make the wheel motion uniform.

B. Model Climbs 20cm Height with Piping at Angle

The simulation model was able to climb the step test of 20cm with two PVC pipes. To do this, a controller had to drive the virtual robot such that it approached the piping at an extreme angle of incidence. The robot would begin to climb up with one track and turn such that the second track would also be on the pipe. Then moving forward it was able to completely pass the step. In testing with the actual robot, the robot would rotate into the direct forward approach when attempting the test at angles. With the robot directly approaching the piping, it spins its tracks and is unable to get on top of the pipes or step. The actual robot was unable to climb the same test height that the virtual model could.

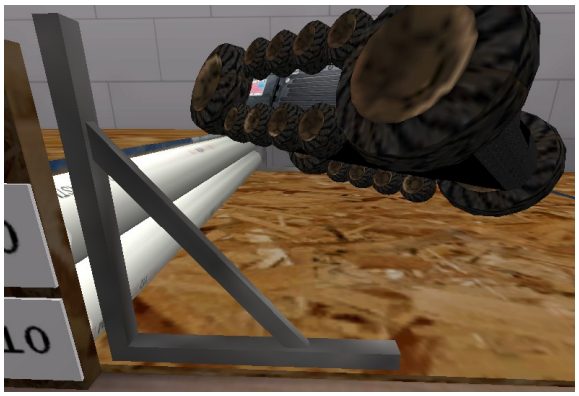

Fig. 10. Talon Model Angled Forward Approach

C. Model Climbs 20cm Height with Piping in Reverse

When the controller drove the Talon in reverse to the step, it was able to rise up the piping and nearly climb the step. The model in USARSim was able to climb the stairs in reverse with ease.

Fig. 11. Talon Model Climbing of Pipe in Reverse

VI. CORRECTING THE ROBOT MODEL

Once a difference between the model and the robot is identified, one can then adjust and fine tune the behavior of the robot model with an understanding of the Karma parameters. This is accomplished by changing the physics and retesting the robot. This process is repeated until a model can be verified through its performance on the test methods.

A. Track Implementation

For the track to behave properly as a group of many wheels, the wheels must all have the same angular velocity. As it is currently, the tires are all spawned by the KDTrack.uc class. Each track spawned is issued commands by USARSim. These commands are then directed to each wheel. When the entire track is issued a command to drive forward, each wheel attempts to do just that. Because the wheels interact differently with the simulation environment, the actual response of the wheel is then calculated on an individual basis. The individual wheels of the track can respond differently to a single command. Another issue was the small wheels needing realistic linear damping to simulate the flex of the track. The KLinearDamping parameter value must be lowered in TalonTrack.uc to produce accurate results.

B. Model Climbs 20cm Height with Piping at Angle

The tracks of the Talon model are able to grip onto the pipe enough to pull the robot on top of the pipe. This is an unrealistic part of the model that produces the uncharacteristic behavior. The classes that control the behavior of the track are the TalonTrackTire.uc and TalonTire.uc. These classes extend KTire, the Unreal Tournament class that characterizes the tires of the Unreal vehicles. Because of this, they inherit control of the friction, slip, and normal properties of the KTire class. In the Talon track classes, the tire properties must be changed to correct the model. The lateral friction on the model is too high if the robot rotates sideways when approaching at high angles of incidence, and the roll friction must be reduced for the track to not be able to grip the pipe. Lastly, the motor torque of the robot must be decreased to lessen its climbing ability. Other parameters such as tire softness, tire adhesion, and the slip rate can affect the performance of the track. These changes have proven to successfully correct the behavior of the robot in testing.

C. Model Climbs 20cm Height with Piping in Reverse

Some of the above parameters that changed with the adjustments on the track will help lessen the problem of the simulated robot climbing the step test in reverse. The class that defines the behavior of the Talon is Talon.uc. The actual robot not being able to climb forward but able to in reverse indicates a center of mass that is not at the center of the robot. The Karma parameters of an actor are defined within the KParams of that object. The property that will change the center of mass is the KCOMOffset, which has not been set in the current model. The center of mass is defaulted to the origin of the robot. By measuring the actual robot to find its center of mass, the KCOMOffset can be accurately changed to be accurate. Should the robot be unavailable for measurement, it is reasonable to estimate the center of gravity from the location of the heavy battery packs in the front of the vehicle. The offset would be near halfway toward the front of the vehicle. This assumption proved accurate in final testing of the behavior of the modified robot model.

VII. MODEL VERIFICATION

The test methods are not only used to alert researchers of physics problems, but are also used to show that a model is accurate. With several of the same test methods, testing revealed that the robot model behaved as the actual robot.

A. Directed Perception

The arm and manipulator control of the Talon are accurately replicated for the Talon model in USARSim, which is illustrated by the directed perception test. The Talon manipulator uses joint level control to move each link individually. The performance data captured in simulation shows a close correspondence to data captured on the actual test method. The range of motion for each joint has been set to realistic values that may be inaccurate to the actual range of motion for the Talon manipulator. This can be corrected after a few tests with the actual robot manipulator.

B. Grasping Dexterity

The manipulator of the Talon was shown to be accurately modeled in the grasping dexterity test. This test also analyzed the gripper of the Talon arm. The robot has a gripper with two fingers, and the model has these at the correct dimensions. The control of the manipulator and gripper have been correctly modeled.

Fig. 12. Talon Model Successful Directed Perception Test

C. Stairs

At a Department of Homeland Security (DHS) workshop held in Las Vegas in 2005, the Talon robot was recorded as it climbed an open stairway with railings. The robot was able to use rocks at the base of the stairs to get on top of the first stair. Once on the stair, it was able to climb the remaining stairs with relative ease. In simulation, the robot had difficulty getting onto the first stair without a small obstacle. With that obstacle in place, the model was able to complete the test with ease[7]. The stairs used in the simulated test have a slope of exactly 40°. This is a slope close in value to that of the stairs on which the Talon was tested, which are estimated at 41°. Both tests were also performed on open stairs.

D. Step Field Pallet

The step field pallets were also useful in verifying the robot model. The robot can perform well on half step field pallets (medium mobility difficulty). The full step field pallets (difficult mobility) however proved challenging for the robot. The actual robot is able to eventually complete the difficult mobility test by reversing and reattempting at different angles, which is also the case for the model robot in USARSim. The model completed the medium mobility test with little trouble, and completed the difficult mobility test with some difficulty.

E. Mobility and Endurance (Zigzag and Figure 8)

Being based on the step field pallets, figure 8s and zigzags highlight much of the same abilities of the robot. Because the medium difficulty mobility is not challenging for the Talon, this test analyzes the endurance of the robot. The difficult tests focus on the mobility of the robot. The battery life of the Talon robot is near four hours at typical operational speed [15]. The battery life of any robot in USARSim is a configurable variable, which defaults to 20 minutes. The battery life is not calculated based on the use of the electric devices or energy consumption of the motors; however, this model simplification is acceptable because the difficulty of implementation outweighs the minimal benefits of battery accuracy. In addition to this, robots in USARSim cannot be damaged yet. Robot damages is being researched, and will be tested with these endurance tests once implemented.


Fig. 13. Talon Robot Pass Stair Test at 2005 DHS Workshop, Las Vegas

VIII. CONCLUSION

This testing has revealed the test methods to be an excellent solution to the problem of determining and increasing simulation accuracy. The simplicity of the tests makes model fabrication and physical construction easier. The test methods created for the ASTM standard test specific characteristics of the robot, making them easy to use for modifying robot models. In using the test methods, a researcher is able to identify a specific problem, and can then improve the model accordingly. Developers can also use these tests to validate existing models, and show that the behavior is accurate to reality.

IX. FURTHER RESEARCH

The changes discussed on the Talon model will be implemented, including forcing the tires of the track to spin at the same angular velocity. NIST is currently investigating possible solutions to the issue. The release of the new Unreal Engine 3 may provide an answer.

Another area of future experimentation is the gripper behavior. Testing will be performed to determine the accuracy of the gripper strength. The arm must also be tested to determine the amount of weight it can lift. A simple test of lifting increasingly heavier weights with the actual Talon in a repeated manner will illustrate the behavior the model should mimic. Repeating the same test in simulation will allow for precise retuning of the model physics. In addition, other commercial platforms will be subjected to similar tests and have their models validated.

7 Because the stairs of the test in Las Vegas are unavailable for friction experimentation, analysis of captured performance data was used to validate the simulated test method. Creating the test method as a static mesh produced results with strong correlation to the robot behavior observed at the DHS Workshop.

As different waves of first responder requirements are implemented in the robots and robot models, more test methods will need to be developed. The individual capabilities of the robot must be tested to ensure that they were implemented correctly.

ACKNOWLEDGEMENTS

The authors would like to thank the developers of USARSim, Benjamin Balaguer for assisting in Talon model modifications, and Tony Downs for conducting and assisting in testing with the Talon robot.

REFERENCES

[1] Messina, E., "Performance Standards for Urban Search and Rescue Robots," ASTM International Standardization News, August 2006.
[2] A. Jacoff, E. Messina, J. Evans, "Performance Evaluation of Autonomous Mobile Robots", Industrial Robot: An International Journal, Volume 29, Number 3, 2002, pp. 259-267.
[3] Epic games, "Unreal engine," 20 Jul 2007. www.epicgames.com, 2005.
[4] M. Lewis, J. Jacobson, "Game engines in scientific research," Communications of the ACM, Volume 45, Number 1, pp. 27–31, 2002.
[5] Gamebots, 20 Jul 2007. http://gamebots.sourceforge.net/.
[6] S. Carpin and M. Lewis and J. Wang and S. Balakirsky and C. Scrapper, "USARSim: a Robot Simulator for Research and Education", Proceedings of the IEEE 2007 International Conference on Robotics and Automation, October 2007.
[7] S. Carpin and J. Wang and M. Lewis and A. Birk and A. Jacoff, "High Fidelity Tools for Rescue Robotics: Results and Perspectives", Robocup 2005: Robot Soccer World Cup X, Springer, LNAI, Volume 4020, 2005, pp. 301-311.
[8] J. Wang and M. Lewis and S. Hughes and M. Koes and S. Carpin, "Validating USARSim for Use in HRI Research", Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting(HFES05), 2005, pp. 457-461.
[9] M. Zaratti and M. Fratarcangeli and L.Iocchi, "A 3D Simulator of Multiple Legged Robots based on USARSim", Robocup 2006: Robot Soccer World Cup X, Springer, LNAI, 2006.
[10] M. Balakirsky, C. Scrapper, S. Carpin, and M. Lewis, "USARSim: A RoboCup Virtual Urban search and Rescue Competition", Proceedings of SPIE, 2007.
[11] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, C. Scrapper, "USARSim: A Robot Simulator for Research and Education", Proceedings of the IEEE International Conference on Robotics and Automation, 2007.
[12] "Unreal Developer Network Karma Reference." Unreal Developer Network. 19 Jul 2007 http://udn.epicgames.com/Two/KarmaReference.html.
[13] "Static Mesh" Wikipedia, the Free Encyclopedia, July 31, 2007, http://en.wikipedia.org/wiki/Static_Mesh.
[14] Busby, Jason, and Zak Parrish. Mastering Unreal Technology: The Art of Level Design. Indianapolis, Indiana: Sams Publishing, 2005.
[15] "Products & Talon Military Robots, EOD, SWORDS, and Hazmat Robots." Foster-Miller, Inc - QineticQ North America. July 13, 2007 http://www.foster-miller.com/lemming.htm.
[16] Jacoff, A., Messina, E., Weiss, B.A., Tadokoro, S., Nakagawa, Y., "Test Arenas and Performance Metrics for Urban Search and Rescue Robots," Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, NE, October 27-31, 2003.
[17] Jacoff, A. and Messina E., "Urban Search and Rescue Robot Performance Standards: Progress Update," Proceedings of the 2007 SPIE Defense and Security Symposium Unmanned Systems Technology IX, Orlando, FL, April 2007.
[18] "teleMAX" telerob, July 23, 2007, http://www.telerob.com.