Published in final edited form as:
Ind Rob. 2016 ; 43(5): 563–572. doi:10.1108/IR-01-2016-0032.

Test Methods for Robot Agility in Manufacturing

Anthony Downs,
Intelligent Systems Division, National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899

William Harrison, and
Intelligent Systems Division, National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899

Craig Schlenoff
Intelligent Systems Division, National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899

Abstract

Purpose—The paper aims to define and describe test methods and metrics to assess industrial robot system agility in both simulation and in reality.

Design/methodology/approach—The paper describes test methods and associated quantitative and qualitative metrics for assessing robot system efficiency and effectiveness which can then be used for the assessment of system agility.

Findings—The paper describes how the test methods were implemented in a simulation environment and real world environment. It also shows how the metrics are measured and assessed as they would be in a future competition.

Practical Implications—The test methods described in this paper will push forward the state of the art in software agility for manufacturing robots, allowing small and medium manufacturers to better utilize robotic systems.

Originality / value—The paper fulfills the identified need for standard test methods to measure and allow for improvement in software agility for manufacturing robots.

1 Introduction

In today’s world of fast-paced change and ever-evolving technologies, there is constant pressure on manufacturers for agile production. The need for customization and robustness requires today’s manufacturers to shift production goals quickly, and address process problems and anomalies in a fluid and efficient manner. Though these pressures can be partially addressed at any part of the supply chain, industrial robotics is an obvious choice

Certain commercial companies, products, and software are identified in this article to explain our research. Such identification does not imply recommendation or endorsement by NIST, nor does it imply that the companies, products, and software identified are necessarily the best available for the purpose.

for addressing process agility. A robot's inherent flexibility and adaptability through programming and sensor implementation make it a good choice for addressing agility.

In the context of this paper, we define agility as "the ability of a robot system to succeed in an environment of continuous and unpredictable change by reacting efficiently and effectively to changing factors." While there is no agreed upon definition of robot agility in the literature, this definition is consistent with proposed definitions of agile manufacturing, which involves not only robot agility but also agility of the manufacturing process as a whole.

Agile manufacturing aims at addressing continuous and unpredictable customers' needs by rapidly and effectively responding to changes in customers' requirements. It demands a manufacturing system that is able to produce effectively a large variety of products and to be reconfigurable to accommodate changes in the product mix and product designs as described by Gunasekaran (1999). Reconfigurability of manufacturing systems, product variety, velocity, and flexibility in production are critical aspects to achieving and maintaining competitive advantage. Agile manufacturing mainly represents the idea of "speed and change in business environment" (Hosseini and Kiarazm, 2014) and consists of two main factors:

1. Responding to change in product development and delivery times, and
2. Exploiting changes and taking advantage of them as opportunities (Sharifi and Zhang, 1999)

There are many ways robot agility challenges could occur on a factory floor, including (in no particular order):

1. The need to swap robots in and out without introducing extended downtimes or reprogramming,
2. The need for a robot to replan if a failure happens (e.g., a part is dropped),
3. The need for a robot to replan when a new goal (order) is provided to it, or
4. The need for a robot to respond to changing environmental conditions (e.g., non-fixtured tray moves).

Beyond enumerating the types of agility challenges that can occur, we also need a way to measure the agility of a robot system. Based on the definition above, we start with a measure of the effectiveness and efficiency of the robot(s) in completing a task. To measure effectiveness, we consider quantitative metrics such as the distance between a part's actual final location and the desired final location (goal location), and for efficiency we consider metrics such as the total time it takes for the system to finish its task. These metrics are compared between a task with an agility challenge and a parallel task without an agility challenge (baseline). An example of this is a material handling task that never drops a part (baseline) compared to the same material handling task that always drops a part. The baseline can be compared to the agility challenge to begin to assess the agility of the system.

The purpose of this paper is to define and describe test methods and metrics to assess industrial robot system agility in both simulation and in reality. The test methods include both quantitative and qualitative metrics for assessing robot system efficiency and effectiveness which can then be used for the assessment of robot system agility. Section 2 contains a background of agility in manufacturing. The test methods are then put forth in Section 3, followed by their implementation in simulation in Section 4. A description of how the test methods can be applied on a real system are put forth in Section 5 followed by a test case given in Section 6. Finally the future work and conclusions can be found in Section 7.

2 Background

2.1 Prior Work in Test Methods

The Intelligent Systems Division (ISD) in the Engineering Laboratory (EL) at the National Institute of Standards and Technology (NIST) has well over 10 years of active efforts in developing test methods. These test methods are being leveraged for this effort of developing test methods for measuring agility in manufacturing robots. Some examples of these are described below.

2.1.1 DHS US&R ASTM Test Methods—Researchers at NIST have been developing standard test methods for evaluating Urban Search & Rescue (US&R) Robots for over 10 years with ASTM International and the Department of Homeland Security (DHS). These test methods are developed with initial and ongoing input and comments coming from multiple user groups, including the first responder community as the "end-users" and robot developers, ensuring a well-balanced set of priorities and ideas for the test methods and what/how they should be testing. The group has developed over 40 test methods by taking the complex tasks that robots need to do, breaking them down into components, and turning those component pieces into test methods. The test methods and development methodology have also been used to cover bomb disposal and military robotics applications as described in Messina (2007).

2.1.2 SCORE—NIST researchers developed a system called SCORE (System, Component, and Operationally-Relevant Evaluation) for evaluating emerging intelligent systems (not limited to robotics). This system was developed because of the need to test complex systems that cannot be tested completely as a full system, but where the individual components that make up the system being tested do not fully test the system either as described in Weiss and Schlenoff (2008). This SCORE system has been applied and used to evaluate soldier-worn sensor systems, automated voice translation systems, and tactical military applications on Android-based handheld devices as described in Schlenoff et al. (2007, 2009); Weiss et al. (2013).

2.2 Other Work in Manufacturing Agility

A literature search for use cases for addressing robot agility was conducted in the earlier stages of the test method development. A few papers that were found are described here for assembly-type manufacturing use cases and change cases. For the assembly-type use cases, Quinn et al. (1997) describe an example assembly task with four plastic parts that get snapped and inserted together. This is described as a typical light assembly task for the workcell being tested, which includes two robots working together. Frei et al. (2008) describe an example assembly of an adhesive tape roll dispenser assembly, which is a slightly more complicated assembly than the first use case as it requires a screw for locking the pieces together. Frei et. al also described a change case in the assembly of the adhesive tape roll dispenser assembly, where the environment gets changed to a different locking method for the assembly process, in this case, changing from a screw-lock assembly method to a snap-fit method of assembly. Another change use case was described by Gou et al. (1994), wherein three cases are described. The first case is used as a baseline to compare performance for the other two cases. The second case is a new high priority order coming into the system to invoke a re-prioritization. The third case is a variation on the second, but the reprioritization is caused in this case by a machine breakdown, causing the system to adjust to absorb the workload.

3 Test Methods Addressing Agility for Manufacturing Robots

In this section we discuss each of the currently developed test methods, including a description of the tasks, apparatus, and metrics to be collected for each test method. The apparatuses for these test methods are a repeatable and reproducible physical or simulated representation of the task to be completed by the robot system. The apparatuses are intended to be relatively inexpensive and use readily available materials in order to allow wide proliferation of the test methods for the purposes of practice. The initial three test methods described below are based on a simplified form of assembly tasks known as kit building (or “kitting”). These kits are the step before the actual assembly process, where the kit that is built contains all the parts that will be needed for the assembly. A kit is defined by Bozer and McGinnis (1992) as a “specific collection of components and/or subassemblies that together (i.e., in the same container) support one or more assembly operations for a given product or shop order.”

3.1 Baseline Kit Building

Since the overall purpose of this suite of test methods is to measure how well the software controlling a manufacturing robot can handle changes, whether external or internal, there is a need for a baseline test method against which a particular system’s performance and handling of the agility challenges can be compared. Since each robotic system has a different set of capabilities and proficiency with the tasks it can perform, the kit building test method provides a baseline to compare against for each system under test. In this way, once the agility challenges are introduced below in the Dropped Part and In Process Kit Change Test Methods, comparisons can be made between the metrics obtained in this Baseline Test Method and the metrics in the Agility Challenge Test Methods. This provides additional comparisons between different robotic systems under test in addition to the straight comparisons of Agility Challenge Test Method Metrics to systems under test.

The baseline test method chosen here is a set of simple kit building tasks, where the robotic system needs to pick up and move pieces from the environment to specific locations and orientations defined by a provided goal file.

3.1.1 Test Method Tasks—The task for the Baseline Kit Building Test Method at the highest level is to build a kit as specified in the goal file given to the System Under Test. A kit defined in a goal file will have a set of parts that need to be placed in the kit tray. The parts that are in the kit can be chosen to be specific to a particular user's needs if that user wants to compare some number of robotic systems to see how well the systems will work in their particular use cases or scenarios. For demonstrative purposes here, two generic examples of kits will be described below. The first generic kit is the Gearbox Kit, containing three gears of different sizes (small, medium, and large) and two pieces of a shell that will encompass the gears in the eventual assembly. (Figure 1)

For this example kit, the tasks for the Kit Building Baseline Test Method are to pick up each of the five parts from their initial locations (often in parts bins or in a parts tray) and then place the individual parts in the kit tray at a specified position and orientation. Once the kit tray is completed (all five parts in their proper place and orientation and only those parts in the kit tray), then the kit tray would be sent off to another part of the factory to be assembled and the robotic system would begin work on a new kit. These follow-on actions are outside the scope of these test methods.

The second generic kit is a set of differently shaped and colored pegs as shown in Figure 2. The tasks involved in this kit are very similar to the Gearbox Kit with the only difference being that the parts being picked up and placed in the kit tray are the shaped pegs. This kit provides for more flexibility in the particular design of the goal file in that there are nine different shapes of pegs to choose some subset of to be in the desired kit. Using this set of parts for the kit tray also lends itself toward a future version or variant of this test method where the tasks would be to pick up the pegs from a surface or parts bin and insert them into a block with shaped holes to be more along the lines of assembly.

3.1.2 Apparatus—The apparatus for the test method can be set up either as a physical, real-world space or in a simulated, virtual world. The physical version of the test method apparatus consists of a tabletop surface with a size of at least 1.5 m × 1.5 m. This tabletop will have a space set aside on it for the "Kit Tray" area as well as a space for the parts to be laid out or parts bins or parts trays if those are used.

This apparatus (and those for the other test methods described below) can be easily scaled to match the size of different robots and/or capabilities (adjusting size of parts, kit tray, etc., as well as weights or shapes of objects depending on the robot's abilities).

3.1.3 Metrics—The metrics that are used and/or calculated for the Baseline Kit Building Test Method are divided up into time metrics, distance metrics, kitting process completion metrics, and failure statistics metrics. For all of the numerical metrics, a lower score is better than a higher score. A summary of the metrics is shown in Table 1.

The time metrics consist of individual task times $\left(T_{Task_i}\right)$, the total time to assemble the entire kit (T_{Total}), and the planning time ($T_{Planning}$). The individual task times $\left(T_{Task_i}\right)$ measure the time that it takes to perform each task i, in this case, moving each part from the initial location to the kit tray. The times start when the part is first picked up by the robot and end when the robot places the part in the kit tray. The total time (T_{Total}) starts when the robot system receives the goal file (which tells the robot what parts to put in the kit tray and where to put them) and ends when the robot signals that it has completed the kit (by sending a "done" message). The planning time ($T_{Planning}$) is an estimate of time the robot system spends planning (and thus not moving parts), and is calculated based on the total time and the sum of the individual task times and is defined as:

$$T_{Planning} = T_{Total} - \sum_{i=1}^{n} T_{Task_i} \quad (1)$$

The planning time can also be expressed as a percentage of the total time:

$$T_{Planning\%} = \frac{T_{Total} - \sum_{i=1}^{n} T_{Task_i}}{T_{Total}} * 100\% \quad (2)$$

The distance metrics consist of total distance traveled $\left(Dist_{Goal_i}\right)$ by each of the parts described in the goal file (goal objects) and the total distance traveled by the robot's end-effector or manipulator ($Dist_{Manip}$). $Dist_{Goal_i}$ is the total distance (in meters), including redundant and repetitive motions, which each goal object moves during the test. So, a part that moves on a winding path from point A to point B will have a longer distance than the same part that moves on a straight path from point A to point B. This distance is recorded for each part in the goal file. The distance is calculated in three dimensions wherever possible or two dimensions (the horizontal plane, ignoring the vertical axis) if three dimensional part tracking is not achievable. The total distance traveled by the end-effector/ manipulator ($Dist_{Manip}$) is, similarly, the total distance (in meters), including redundant and repetitive motion, which the robot's end-effector or manipulator moves during the entire test method time period (as measured by T_{Total}). Again, this distance will be calculated in three dimensions wherever possible or two dimensions (the horizontal plane, ignoring the vertical axis) if three dimensional part tracking is not achievable.

The kitting process completion metrics are success metrics for the task and consist of qualitative success metrics for both the individual tasks and for the kit as a whole, quantitative success for position and rotation for each task and the kit as a whole, and the total number of attempts required to complete each task. The qualitative task level success metric is a binary yes/no value describing for each task whether the task was completed successfully (i.e., the part was placed in the kit tray). The qualitative kit level success metric is the binary yes/no value for whether the kit is complete (i.e., all the required parts, and no other parts, are in the kit tray). Note, for each of these qualitative success metrics, the positional and rotational accuracy of the parts within the kit tray are not taken into account. As long as the part is somewhere in the kit tray, these metrics would read as "yes." The quantitative positional task level success metric is the distance (in meters) representing the linear distance between the location of the part associated with the task and the goal location. The quantitative rotational task level success metric is the the rotation (in degrees) between the orientation of the part associated with the task and the goal orientation. These distances and rotations are measured in reference to the centroid of the part.

The quantitative positional kit level success metric is the sum of the individual task level positional success metrics as absolute values:

$$Kit\ Level_{Quantitative} = \sum_{i=1}^{n} \left| Task\ Level_{Quantitative_i} \right| \quad (3)$$

The quantitative rotational kit level success metric follows the same formula (3) but with the individual rotational task level success metrics. For comparative purposes, the rotational error for each part will be reported as a single angle of rotation about a line, in order to be able to better judge the accuracy compared to the desired orientation of the parts within the kit tray. The total number of attempts required to complete each task is a count, for each individual task, of the number of attempts that were needed to complete the task. A new attempt would be counted if, for example, the robot were to drop the part while in the process of moving it to the kit tray.

The failure statistics metrics currently consist of only one metric: the number of failures, which is simply a count of the number of failures that occurred during the test method process. For the purposes of these test metrics, a failure is defined as any situation where a human must intervene and thus the system is no longer autonomous.

Some of the metrics described here are being used in a combined fashion by using a weighted sum of the values to determine the “better” solution for the limited scope of the tasks in a competition that is being planned for next year.

3.2 Dropped Part

The Dropped Part Test Method is the first of the test methods to add an actual agility task into the mix. The idea behind this test method is to determine how well the System Under Test (SUT) can determine successful completion of a task and if the system is aware of the parts as they are being moved from place to place. The system’s handling of this agility obstacle can be compared to the baseline kit building test method above and the performance can be extrapolated to how the system will perform in real-world scenarios.

3.2.1 Test Method Tasks—The tasks for the Dropped Part Test Method are largely similar to the Baseline Kit Building Test Method described in Section 3.1.1, but with the added complication that while the robot is in the process of moving one of the parts, which has a duplicate in the parts storage area, the part is forced to be dropped from the gripper. The robotic system under test is then observed to see:

- Is the SUT is aware of the part having dropped?
- How did the SUT became aware of the part having dropped (this is determined by the SUT and needs to be sent over the interface as further described in Section 4.1)?

- Does the SUT alert the user to the fact that the part was dropped (does it need to?)?
- Does the SUT find and pick up the part that was dropped to put it in the kit tray?
- Does the SUT pick one of the duplicate parts from the parts storage area to put it in the kit tray?

3.2.2 Apparatus—The apparatus for the Dropped Part Test Method is also largely similar to the Baseline Kit Building Test Method described in Section 3.1.2, but with the additional requirement that at least one part for the kit has to have a duplicate part in the parts storage area. This means that for the Gearbox Kit, one of the five parts (three gears and the two case pieces) will need to have a duplicate. For the shaped pegs kit, the extra parts are already built-in for most cases since each part block in the parts storage area holds nine instances of each part. For both the initial part pickup and the case where the robot picks up a duplicate part for the kit, the SUT can pick up any of the nine pieces available in the block.

3.2.3 Metrics—The metrics for the Dropped Part Test Method (Table 2) are also largely similar to the Baseline Kit Building Test Method described in Section 3.1.3, with the additional metrics for how and if the SUT handles the dropped part. All of the metrics for time, distance, and kitting process completion are the same as the baseline test method, and the failure statistics metric is largely the same as well, but focused more on failures related to the dropped part. For instance, if the SUT fails to realize the part was dropped and does not pick up either the dropped part or a duplicate part to place into the kit tray, then that would be a failure in this test method. There are also additional metrics specific to this Dropped Part Test Method called agility challenge handling metrics. These new metrics include an awareness of the dropped part metric consisting of a yes/no or boolean metric indicating whether the SUT was able to sense or be aware of the part having been "dropped" from its end-effector and a note field regarding how the SUT became aware of the dropped part (e.g., pressure sensors in end-effector registered a drop in pressure? Camera system saw part drop and alerted?). The information to fill in this note field will need to be reported by the SUT through the provided interface. Did the SUT pick the part that was dropped and continue on its way to the kit tray? Or, did the SUT instead decide to pick up one of the duplicate parts from the parts storage area?

3.3 In Process Kit Change

The second agility challenge presented in these test methods is an In Process Kit Change. The idea behind this test method is that while the system is in the process of building a kit, a higher priority order comes in, and the system must figure out how to best handle the new order.

3.3.1 Test Method Tasks—The tasks for the In Process Kit Change Test Method are, again, similar to those of the Baseline Kit Building Test Method described in Section 3.1.1. This time, while the SUT is in the process of building the first kit, and after the SUT has placed at least two parts into the kit tray, a new higher priority kit gets sent in. The robotic system is then observed to see:

- Is the SUT is aware of the new kit order?
- Does the SUT analyze the partial kit tray to see if it can use parts of it for the new kit?
- Does the SUT take parts out of the partial kit tray that are not needed in the new kit?
- Does the SUT set the partial kit aside and start a new kit from scratch?

3.3.2 Apparatus—The apparatus for the In Process Kit Change Test Method is, again, largely similar to the Baseline Kit Building Test Method described in Section 3.1.2, but with the additional requirement that there are enough parts in the parts storage area to fully complete both kits. An example layout using the Gearbox Kit in this test method would be to have the initial kit contain the top, bottom, and the small gear, while the higher priority second kit would contain the top, bottom, large gear, and medium gear. So, the parts storage area would contain, at minimum, a large gear, medium gear, small gear and two instances each of the top and bottom covers. For the shaped colored pegs layout, there will likely be no additional parts needed since this layout has nine of each shaped peg available in the parts storage area.

3.3.3 Metrics—The metrics for the In Process Kit Change Test Method (Table 3) are also largely similar to the Baseline Kit Building Test Method described in Section 3.1.3, with the additional metrics for how the SUT handles the new, higher priority kit. All of the metrics for time, distance, kitting process completion, and failure statistics are the same as the baseline test method. There is also a set of agility challenge handling metrics for this In Process Kit Change Test Method, similar to the Dropped Part Test Method. These include a yes/no metric for whether the SUT scraps the partially built kit (but uses the same kit tray), a yes/no metric for whether the SUT starts over with a new kit tray for the priority kit, a yes/no metric for whether the SUT reuses the parts common between the two kits in building the priority kit, and a yes/no (or n/a) metric for whether the SUT removes the parts not in common between the two kits from the kit tray to build the priority kit. These metrics will be used to separate which trials or instances of a SUT can be accurately compared. These agility challenge handling metrics are all observed with no need for the SUT to report them on the provided interface.

4 Test Methods in Simulation

Robot agility can be pursued from both a hardware and software perspective. The test methods described here lean toward software. This is not to say that hardware does not play a role in robot agility, because it most certainly does. It is useful, however, to measure robot system (robot sensors and surrounding equipment) agility without requiring that all physical machines and equipment be present. This is only possible through simulation.

A simulation world and robot interface were developed for assessing robot system agility as well as automatic reporting of agility metrics. V-REP by Coppelia Robotics (2014) was chosen as a simulation tool because of its multi-platform support as well as its strong support for industrial robotics. Many of the open source options such as Gazebo by OSRF

(2014) were either single platform, or better suited for other robotics fields. Industrial simulators, which potentially suit this application well, can be an order of magnitude more expensive, and oftentimes do not provide the flexibility required. In choosing the simulation tool, it needed to be able to handle custom non-standard control interfaces as well as the ability to change and customize the physics implementation.

4.1 Control Interface

After settling on a software tool, the next important aspect of test methods measurement in simulation is the consideration of how the software will be used. At this juncture it is important to think of the simulation as a Tool for Agility Measurement (TAM). The TAM must then be both reproducible and fair with its metrics and how they are implemented. Fairness means that the TAM should minimize all advantages that may be gained by a user having a better understanding of the simulation software. Agility metrics measurements should not be a function of simulation implementation. Reproducibility can be addressed by making sure all virtual environments between measurements are identical (trivial) and being sure that performance is closely tied to task completion only, and not unforeseen synergies between hardware or software and the measurement tactics. These requirements must apply to both the interface and the virtual environment itself.

The interface between the virtual environment and the SUT(Figure 3) should be open and easily implemented. Ideally the interface should have a documented structure with a clear mode of transport and predetermined message content. Additionally, the interface should be the same as that used by the vendor’s physical robot when it is in place. The landscape of industrial communication, however, makes this a challenging endeavor. Current industrial communication protocols abide by standards, i.e., DeviceNet, Ethernet/IP, or Profibus, however, the message content is not standardized. For a solution that fits the requirements, simple messaging was used. Simple messaging is an industrial protocol developed for the Robot Operating System (ROS) by the ROS-Industrial organization and then extended by NIST to add more potential applications (SWRI and ROS-I, 2014). Simple messaging provides for a standardized content structure for the communication.

Simple messaging serves as a good communication protocol because it allows any controller to command a robot with just a transmission control protocol/Internet Protocol (TCP/IP) socket connection and adherence to the message structure. This has the added benefit of fairness, in that all virtual hardware can be pre-validated and pre-loaded. The TAM can have robots of various makes and models already loaded within it, and if a vendor wants to use another robot, they only need a virtual model with the kinematics and dynamics of the desired robot. The new robot can simply be added and then checked to make sure there is no simulation enabled advantage in the model. Simple messaging also allows the SUT to essentially be a black box where their entire control strategy can remain closed and unknown to the public.

4.2 Virtual Environment

The virtual environment houses the robot’s sensors and all other parts and equipment relevant to the test. Figure 4 shows a screen capture of the TAM environment within V-REP.

Because this is a kitting application, parts are moved from one tray to another, in this case from Tray 1 to Tray 2. Simulating in a virtual space also allows the visualization of non-physical but useful items. The start and end points, for example, are regions of space depicted by green and red spheres respectively. When the robot's blue locating sphere crosses the green sphere, the process timer will start, and when the locating sphere crosses the red sphere it will stop. These spheres are used as an alternate method of starting and stopping the time if not using the "done" messaging system described in Section 3.1.3.

When building the TAM environment, it was important to consider what level of physics fidelity would be necessary. For accurate simulation, having every aspect of the environment represented in full physics would be ideal, however, physics engines may manifest physics artifacts in their implementation. An example of this kind of artifact might be a part moving slowly along the floor, seemingly of its own power, or shooting with high velocity from a gripper. Anomalous behaviors are more likely during gripping. The nature of collision detection and discrete time step simulations, mean that artifacts are more likely when collision is encountered on two opposite sides of the same object simultaneously. For this reason gripping is done through object parenting instead of physics-enabled contact. When the parts are dropped, the parenting is removed and then the physics is allowed to take over. The TAM environment has physics enabled for all other aspects of the environment, including part-to-part and part-to-table.

The trade-off in physics was thought to be warranted because the SUT is being evaluated for system agility and not hardware effectiveness. Evaluating the intelligent of a system is achieved by observing and tracking how it responds to unplanned scenarios. This is not to say that hardware capability is not important; hardware is indispensable, however, it is not a part of the test procedure in the TAM environment. Additionally, the sometimes non-repeatable, unrealistic, anomalies of physics engines would hurt the credibility of the test.

5 Test Methods with the Real System

It is important that the simulated test method procedure be as parallel as possible to the test method procedure on physical hardware. This means that the interface and environment of the SUT in simulation should be the same as that of the SUT that is physically present. This is the goal of any simulation, however, this is not true just for the simulation's virtual enabled environment, but also in the interface and measurement procedures of the TAM itself. The control interface and virtual environment must parallel their real counterparts as much as possible. Achieving this parallelism, however, could hurt the legitimacy of the test.

5.1 Control Interface

Ideally the control interface to the simulated robot and sensors should be the same for every testing scenario regardless of the SUT. This is the reason Section 4.1 proposed using simple messaging. The rationale being that the vendors should not be allowed to optimize the simulation environment for their system. This reason is not present in the real system. It is then unnecessary to force the SUT to use a particular interface, when the reason for using that interface is no longer present. This difference in requirements between the simulated and real testing setups means that agility test metric measurements from the simulated SUT cannot necessarily be extrapolated to the real SUT. In order to extrapolate results from the simulated SUT to the real, the interface must be the same.

In practice, it makes more sense not to prescribe the interface between the controller and the robot when the test is real. Though, as mentioned above, it makes the simulated and non-simulated metric measurements potentially non-parallel, and it does not hurt the relevance in relative measurements between SUTs. Agility metrics measured in simulation can be considered as an appraisal of agility strategy, and not agility implementation necessarily. Vendors and process designers can use agility metric measurements to design the real agile process.

5.2 Test Environment

The testing environment of the real SUT is in principle identical to that of the simulated. Metrics in this case come from three dimensional tracking provided externally. Algorithms run on the real time tracking information to enable agility measurements. Test situations like dropping a part in the real SUT must be simulated, in that the circumstance must be forced.

6 Test Method Implementation

Though the test methods have been described in detail thus far, and their actual implementation can be carried out in many different ways, describing one such implementation will provide clarity in their understanding. Here the authors describe how a TAM may be applied to a simulated SUT. This procedure can be carried out by a neutral third party or the SUT developers themselves.

The assessment process starts with a virtual environment created in V-REP. The environment consists of a KUKA LWR 4+ and gripper with a number of parts and two trays on a table (Figure 4). The goal is for the robot to move the six parts from Tray 1 to the empty tray (Tray 2). At this point, the virtual environment can be completely provided by a third party. The controllers for the SUT connect to the environment via the simple messaging interface described in Section 4.1. There are separate simple messaging connections for controlling the robot and the gripper. There is also another simple messaging interface for sensor information that tells the control system the location of each part. Utilizing these simple messaging interfaces draws a clear line between the internals of the simulation and the controlling part of the SUT, making it much harder to cheat the test methods.

The SUT begins its baseline process task by moving the locating sphere/gripper through the start sphere, at which point the TAM begins tracking all pertinent quantitative metrics. Upon the completion of the kitting task, the SUT will then move the locating sphere/gripper of the robot to the stop sphere. After the simulation is complete, the report in Figure 5 is auto-generated to display the results.

There are seven sections to the document, each of which is described below.

Header: The top section is the header of the document and has all the general information. This header information is either automatically inferred from the test setup or read from the test form (Figure 6).

Traversed distance: The total distance traversed by the part. Even if a part returns to its original location, traversed distance will read the distance the part has traveled.

Resultant travel distance: The translational distance between the part's starting and ending location.

Distance from goal position: The difference in position between the part's goal position and its final position.

Rotation from goal orientation: The difference in rotation between the part's goal orientation and its final orientation.

Time in gripper: The amount of time the part spends in the gripper.

Process time: The total time for the process. Time starts when the locating sphere crosses the green start sphere (see Figure 4), and the time stops when the locating sphere crosses the red stop sphere.

After the SUT completes the baseline task, the SUT again attempts to do the same kitting task as the baseline, only this time the interface in Figure 6 is used to force the gripper to drop the part. The SUT will then handle this aberration the best way it can. Meanwhile, as with the baseline, the TAM will assess all of the quantitative metrics. When the task is complete, the SUT will signal its completion in the same manner as in the baseline, at which point another auto-generated report will be created. After both tasks have been completed, the metrics described in Section 3 can be calculated and reported.

7 Conclusions and Future Work

The plan for the three test methods described in this paper is to submit them to an international standards body as draft proposals for further development. The authors are currently evaluating which particular standards body they will fit best in. Regardless of which standards body is ultimately chosen, sample data needs to be gathered by running representative robotic systems through the test methods and calculating the metrics to ensure that the metrics and data being gathered will properly assess the full field of available robots (i.e., the tasks/metrics for the robots are not all too simple or too difficult). This sample data will also be used to ensure that the data and analysis that can be achieved provides useful information to potential end-users. Once the data is gathered and analyzed and the standards body has been chosen, the test methods will be put forth as a baseline starting point for standardization.

The current plan for the next test methods that will be developed includes the following ideas. The first idea is a Moved Part During Operation Test Method, where a part needed for a kit is moved from its initial location in the parts storage area to a different location while the SUT is moving a different part to the kit tray area. The next idea is a Missing Part Test Method, where at least one part that is needed for the kit is missing from the parts storage area and thus, the kit cannot be completed. The third idea is a Mismatched Weight of Part Test Method, where a part that is needed for the kit is inexplicably heavier than expected to the point that the system cannot pick it up to move it to the kit tray. The authors are also considering further ideas to be made into test methods.

There is also potential for a robot competition using these test methods and metrics for scoring. This potential competition is still in an early planning stage. More details will be made available when determined.

NIST has developed three draft test methods for measuring software agility for manufacturing robot systems, with a set of metrics designed to allow comparisons between robot systems in changing environments. Two test methods, to show baseline and changed performance when changes are introduced in the environment, have been implemented in a simulation environment with semi-automated metrics and a report generated when the test methods are run with a simulated robotic system.

References

Bozer YA, McGinnis LF. Kitting versus line stocking: a conceptual framework and a descriptive model. International Journal of Production Economics. 1992; 28(1):1–19.

Coppelia Robotics. Virtual Robot Experimentation Platform (V-REP). 2014. [online] Coppelia Robotics. Available from http://www.coppeliarobotics.com [Accessed 10 March 2014]

Frei, R., Di Marzo Serugendo, G., Barata, J. Self-Adaptive and Self-Organizing Systems. IEEE; 2008. Designing self-organization for evolvable assembly systems; p. 97-106.Second IEEE International Conference Proceedings SASO'08

Gou, L., Hasegawa, T., Luh, PB., Tamura, S., Oblak, JM. Computer Integrated Manufacturing and Automation Technology. IEEE; 1994. Holonic planning and scheduling for a robotic assembly testbed; p. 142-149.Proceedings of the Fourth International Conference on

Gunasekaran A. Agile manufacturing: a framework for research and development. International Journal of Production Economics. 1999; 62(1):87–105.

Hosseini MH, Kiarazm A. Presenting agile supply chain model by using interpretive structural modeling (ISM) approach. International Journal of Operations and Logistics Management. 2014; 3(4):337–350.

Messina E. Performance standards for urban search & rescue robots: Enabling deployment of new tools for responders. Defense Standardization Program Office Journal. 2007:43–48.

OSRF. Gazebo. 2014. [online] Open Source Robotics Foundation. Available from http://www.gazebosim.org [Accessed 26 February 2014]

Quinn RD, Causey GC, Merat FL, Sargent DM, Barendt NA, Newman WS, Velasco VB Jr, Podgurski A, Jo J, Sterling LS, Kim Y. An agile manufacturing workcell design. IIE Transactions. 1997; 29(10):901–909.

Schlenoff, C., Sanders, G., Weiss, B., Proctor, FM., Steves, MP., Virts, A. Proceedings of the 9th Workshop on Performance Metrics for Intelligent Systems. ACM; 2009. Evaluating speech translation systems: Applying SCORE to TRANSTAC technologies; p. 223-230.

Schlenoff C, Steves MP, Weiss BA, Shneier M, Virts A. Applying SCORE to field-based performance evaluations of soldier worn sensor technologies. Journal of Field Robotics. 2007; 24(8–9):671–698.

Sharifi H, Zhang DZ. A methodology for achieving agility in manufacturing organisations: An introduction. International Journal of Production Economics. 1999; 62(1):7–22.

SWRI and ROS-I. ROS-Industrial. 2014. [online] Southwest Research Institute and ROS-Industrial Consortium Americas. Available from http://rosindustrial.org/ [Accessed 10 March 2014]

Weiss, BA., Fronczek, L., Morse, E., Kootbally, Z., Schlenoff, C. SPIE Defense, Security, and Sensing. International Society for Optics and Photonics; 2013. Performance assessments of Android-powered military applications operating on tactical handheld devices; p. 875504-2-875504-15.

Weiss, BA., Schlenoff, C. Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems. Association for Computing Machinery; 2008. Evolution of the SCORE framework to enhance field-based performance evaluations of emerging technologies; p. 1-8.

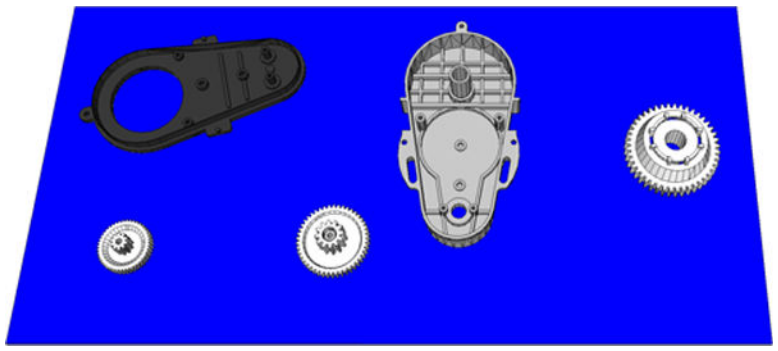


Figure 1.
CAD model showing the parts laid out on a parts surface for the Gearbox Kit.

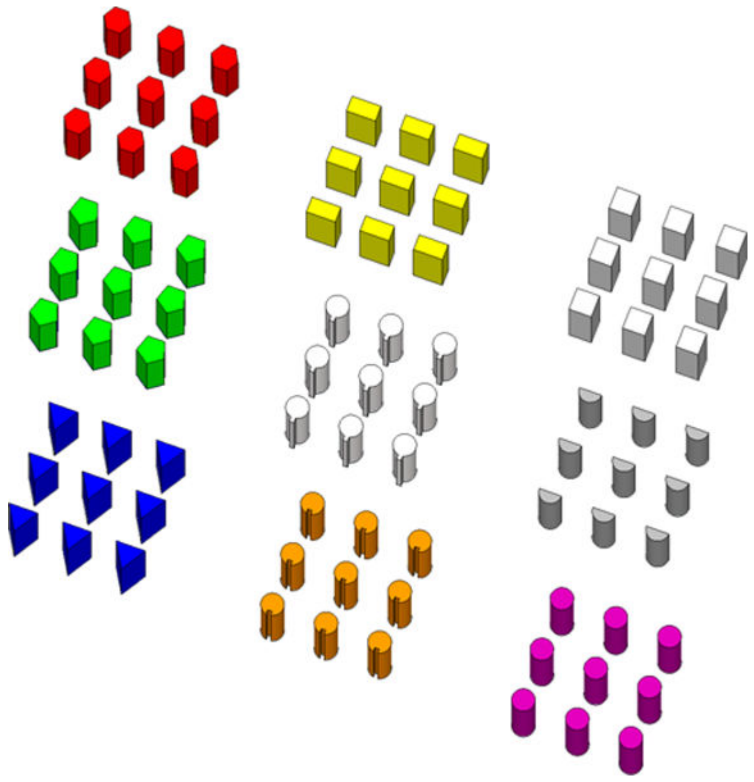


Figure 2.
CAD Model showing the nine different shaped and colored pegs.

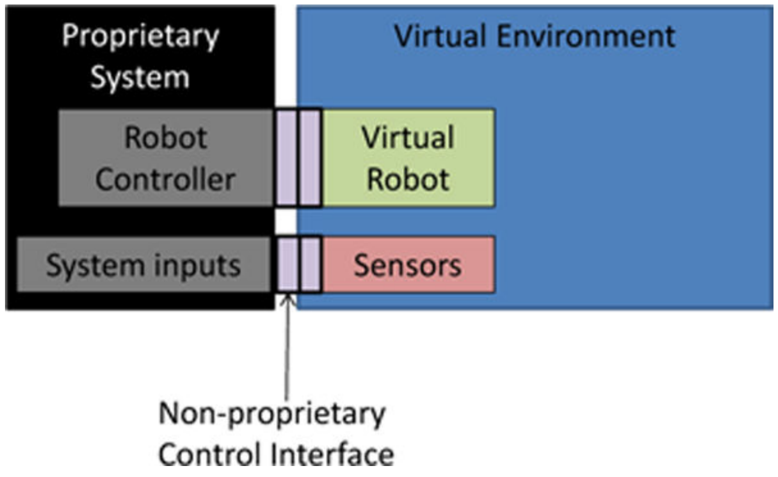


Figure 3.

Diagram illustrating how the user's control system interfaces with the environment.

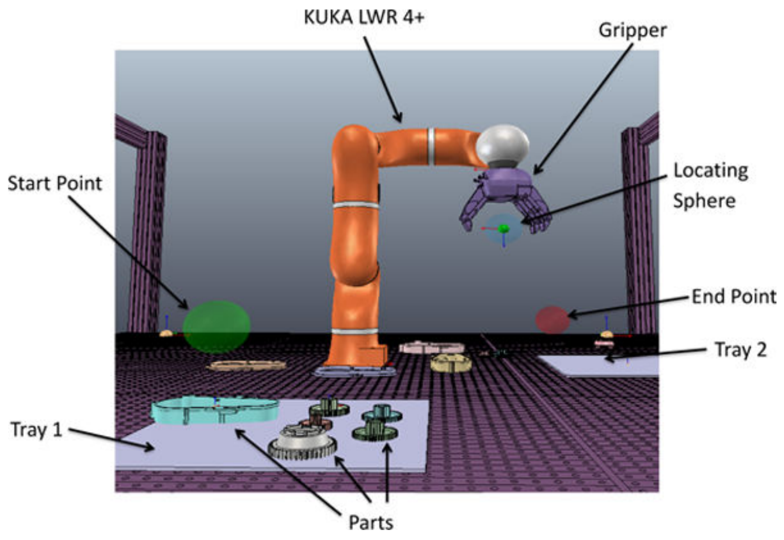
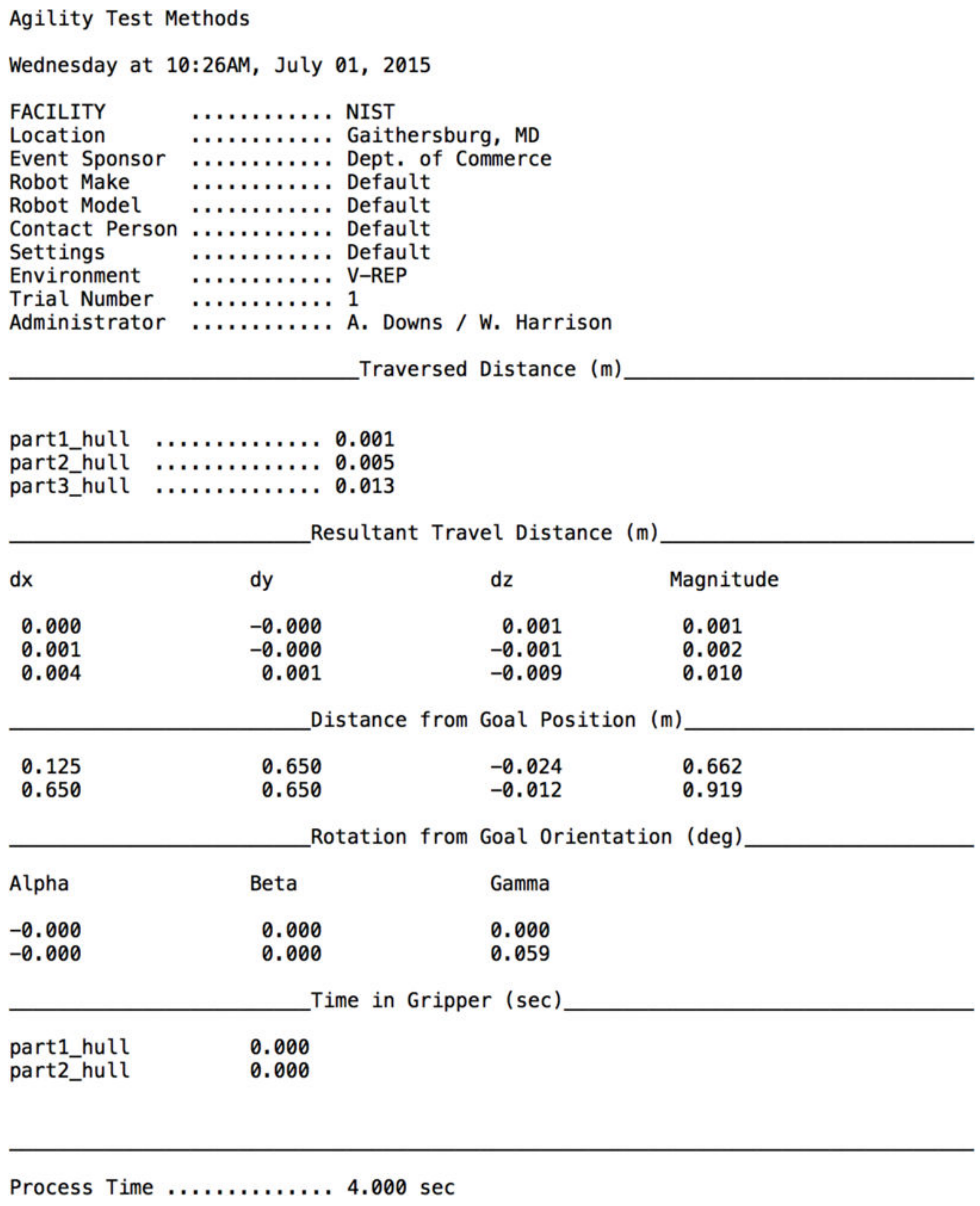


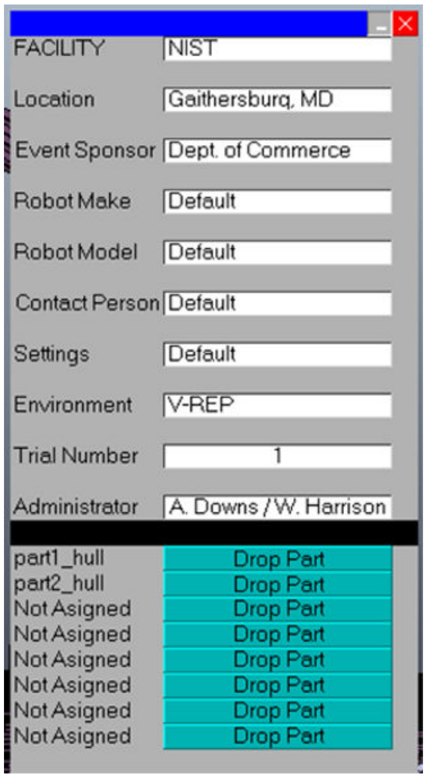
Figure 4.
A screen capture of the V-REP environment.



```
Agility Test Methods

Wednesday at 10:26AM, July 01, 2015

FACILITY       ............ NIST
Location       ............ Gaithersburg, MD
Event Sponsor  ............ Dept. of Commerce
Robot Make     ............ Default
Robot Model    ............ Default
Contact Person ............ Default
Settings       ............ Default
Environment    ............ V-REP
Trial Number   ............ 1
Administrator  ............ A. Downs / W. Harrison
_____________________________Traversed Distance (m)_____________________________


part1_hull  .............. 0.001
part2_hull  .............. 0.005
part3_hull  .............. 0.013

_________________________Resultant Travel Distance (m)__________________________

dx                  dy                  dz              Magnitude

 0.000              -0.000               0.001           0.001
 0.001              -0.000              -0.001           0.002
 0.004               0.001              -0.009           0.010

_________________________Distance from Goal Position (m)________________________

 0.125               0.650              -0.024           0.662
 0.650               0.650              -0.012           0.919

_________________________Rotation from Goal Orientation (deg)___________________

Alpha               Beta                Gamma

-0.000               0.000              0.000
-0.000               0.000              0.059

_________________________Time in Gripper (sec)__________________________________

part1_hull          0.000
part2_hull          0.000


________________________________________________________________________________

Process Time ............. 4.000 sec
```

Figure 5.

Auto-generated report from the TAM.



FACILITY	NIST
Location	Gaithersburg, MD
Event Sponsor	Dept. of Commerce
Robot Make	Default
Robot Model	Default
Contact Person	Default
Settings	Default
Environment	V-REP
Trial Number	1
Administrator	A. Downs / W. Harrison

part1_hull	Drop Part
part2_hull	Drop Part
Not Asigned	Drop Part
Not Asigned	Drop Part
Not Asigned	Drop Part
Not Asigned	Drop Part
Not Asigned	Drop Part
Not Asigned	Drop Part

Figure 6.

Process Form for header information.

Table 1

Baseline Kit Building Test Method Metrics

Baseline Kit Building Metrics		
Category	**Metric Name**	**Units**
Time Metrics	Task Time	(s)
	Total Time	(s)
	Planning Time	(s) or (%)
Distance Metrics	Goal Object Distance	(m)
	Manipulator Distance	(m)
Kitting Process Completion Metrics	Qualitative Task Level Success	(yes/no)
	Qualitative Kit Level Success	(yes/no)
	Quantitative Positional Task Level Success	(m)
	Quantitative Rotational Task Level Success	(degrees)
	Quantitative Positional Kit Level Success	(m)
	Quantitative Rotational Kit Level Success	(degrees)
	Total Number of Attempts	(no unit)
Failure Statistics Metrics	Number of Failures	(no unit)

Table 2

Dropped Part Test Method Metrics

Dropped Part Metrics		
Category	**Metric Name**	**Units**
Time Metrics	Task Time	(s)
	Total Time	(s)
	Planning Time	(s) or (%)
Distance Metrics	Goal Object Distance	(m)
	Manipulator Distance	(m)
Kitting Process Completion Metrics	Qualitative Task Level Success	(yes/no)
	Qualitative Kit Level Success	(yes/no)
	Quantitative Positional Task Level Success	(m)
	Quantitative Rotational Task Level Success	(degrees)
	Quantitative Positional Kit Level Success	(m)
	Quantitative Rotational Kit Level Success	(degrees)
	Total Number of Attempts	(no unit)
Failure Statistics Metrics	Number of Failures	(no unit)
Agility Challenge Handling Metrics	Awareness of Dropped Part	(yes/no)
	Awareness Note Field	(note field)
	Dropped Part Picked Up	(yes/no)
	Duplicate Part Picked Part	(yes/no)

Table 3

In Process Kit Change Test Method Metrics

In Process Kit Change Metrics		
Category	**Metric Name**	**Units**
Time Metrics	Task Time	(s)
	Total Time	(s)
	Planning Time	(s) or (%)
Distance Metrics	Goal Object Distance	(m)
	Manipulator Distance	(m)
Kitting Process Completion Metrics	Qualitative Task Level Success	(yes/no)
	Qualitative Kit Level Success	(yes/no)
	Quantitative Positional Task Level Success	(m)
	Quantitative Rotational Task Level Success	(degrees)
	Quantitative Positional Kit Level Success	(m)
	Quantitative Rotational Kit Level Success	(degrees)
	Total Number of Attempts	(no unit)
Failure Statistics Metrics	Number of Failures	(no unit)
Agility Challenge Handling Metrics	SUT Scraps Partial Kit, Uses 1st Kit Tray	(yes/no)
	SUT Starts With New Kit Tray	(yes/no)
	SUT Reuses Common Parts	(yes/no)
	SUT Removes Non-Common Parts	(yes / no / n/a)