Robotics Developer Studio Reference Platform Design

RDS 4 Beta

September 2011

Microsoft Corporation Technical Documentation License Agreement

READ THIS! THIS IS A LEGAL AGREEMENT BETWEEN MICROSOFT CORPORATION ("MICROSOFT") AND THE RECIPIENT OF THESE MATERIALS, WHETHER AN INDIVIDUAL OR AN ENTITY ("YOU"). BY ACCESSING, USING OR PROVIDING FEEDBACK ON THE ATTACHED MATERIALS, YOU AGREE TO THESE TERMS.

1. For good and valuable consideration, the receipt and sufficiency of which are acknowledged, You and Microsoft agree as follows:

(a) If You are an authorized representative of the corporation or other entity designated below ("**Company**"), and such Company has executed a Microsoft Corporation Non-Disclosure Agreement that is not limited to a specific subject matter or event ("**Microsoft NDA**"), You represent that You have authority to act on behalf of Company and agree that the Confidential Information, as defined in the Microsoft NDA, is subject to the terms and conditions of the Microsoft NDA and that Company will treat the Confidential Information accordingly;

(b) If You are an individual, and have executed a Microsoft NDA, You agree that the Confidential Information, as defined in the Microsoft NDA, is subject to the terms and conditions of the Microsoft NDA and that You will treat the Confidential Information accordingly; or

(c)If a Microsoft NDA has not been executed, You (if You are an individual), or Company (if You are an authorized representative of Company), as applicable, agrees: (a) to refrain from disclosing or distributing the Confidential Information to any third party for five (5) years from the date of disclosure of the Confidential Information by Microsoft to Company/You; (b) to refrain from reproducing or summarizing the Confidential Information; and (c) to take reasonable security precautions, at least as great as the precautions it takes to protect its own confidential information, but no less than reasonable care, to keep confidential the Confidential Information. You/Company, however, may disclose Confidential Information in accordance with a judicial or other governmental order, provided You/Company either (i) gives Microsoft reasonable notice prior to such disclosure and to allow Microsoft a reasonable opportunity to seek a protective order or equivalent, or (ii) obtains written assurance from the applicable judicial or governmental entity that it will afford the Confidential Information the highest level of protection afforded under applicable law or regulation. Confidential Information shall not include any information, however designated, that: (i) is or subsequently becomes publicly available without Your/Company's breach of any obligation owed to Microsoft; (ii) became known to You/Company prior to Microsoft's disclosure of such information to You/Company pursuant to the terms of this Agreement; (iii) became known to You/Company from a source other than Microsoft other than by the breach of an obligation of confidentiality owed to Microsoft; or (iv) is independently developed by You/Company. For purposes of this paragraph, "Confidential Information" means nonpublic information that Microsoft designates as being confidential or which, under the circumstances surrounding disclosure ought to be treated as confidential by Recipient. "Confidential Information" includes, without limitation, information in tangible or intangible form relating to and/or including released or unreleased Microsoft software or hardware products, the marketing or promotion of any Microsoft product, Microsoft's business policies or practices, and information received from others that Microsoft is obligated to treat as confidential.

2. You may review these Materials only (a) as a reference to assist You in planning and designing Your product, service or technology ("Product") to interface with a Microsoft Product as described in these Materials; and (b) to provide feedback on these Materials to Microsoft. All other rights are retained by Microsoft; this agreement does not give You rights under any Microsoft patents. You may not (i) duplicate any part of these Materials, (ii) remove this agreement or any notices from these Materials, or (iii) give any part of these Materials, or assign or otherwise provide Your rights under this agreement, to anyone else.

3. Your right to use these Materials, as described herein, do not change the terms that apply to your use of other materials, software or devices (such as the Kinect device) or provide you any additional rights to modify other materials, software or devices.

4. These Materials may contain preliminary information or inaccuracies, and may not correctly represent any associated Microsoft Product as commercially released. These Materials are not designed or intended to be used in any other product or application that can lead to death, personal injury, property or environmental damage. You assume the sole risk and liability of any use of these Materials. All Materials are provided entirely "AS IS." To the extent permitted by law, MICROSOFT MAKES NO WARRANTY OF ANY KIND, DISCLAIMS ALL EXPRESS, IMPLIED AND STATUTORY WARRANTIES, AND ASSUMES NO LIABILITY TO YOU FOR ANY DAMAGES OF ANY TYPE IN CONNECTION WITH THESE MATERIALS OR ANY INTELLECTUAL PROPERTY IN THEM.

5. If You are an entity and (a) merge into another entity or (b) a controlling ownership interest in You changes, Your right to use these Materials automatically terminates and You must destroy them.

6. You have no obligation to give Microsoft any suggestions, comments or other feedback ("Feedback") relating to these Materials. However, any Feedback you voluntarily provide may be used in Microsoft Products and related specifications or other documentation (collectively, "Microsoft Offerings") which in turn may be relied upon by other third parties to develop their own Products. Accordingly, if You do give Microsoft Feedback on any version of these Materials or the Microsoft Offerings to which they apply, You agree: (a) Microsoft may freely use, reproduce, license, distribute, and otherwise commercialize Your Feedback in any Microsoft Offering; (b) You also grant third parties, without charge, only those patent rights necessary to enable other Products to use or interface with any specific parts of a Microsoft Product that incorporate Your Feedback; and (c) You will not give Microsoft any Feedback (i) that You have reason to believe is subject to any patent, copyright or other intellectual property claim or right of any third party; or (ii) subject to license terms which seek to require any Microsoft Offering incorporating or derived from such Feedback, or other Microsoft intellectual property, to be licensed to or otherwise shared with any third party.

7. Microsoft has no obligation to maintain confidentiality of any Microsoft Offering, but otherwise the confidentiality of Your Feedback, including Your identity as the source of such Feedback, is governed by Your NDA.

8. This agreement is governed by the laws of the State of Washington. Any dispute involving it must be brought in the federal or state superior courts located in King County, Washington, and You waive any defenses allowing the dispute to be litigated elsewhere. If there is litigation, the losing party must pay the other party's reasonable attorneys' fees, costs and other expenses. If any part of this agreement is unenforceable, it will be considered modified to the extent necessary to make it enforceable, and the remainder shall continue in effect. This agreement is the entire agreement between You and Microsoft concerning these Materials; it may be changed only by a written document signed by both You and Microsoft.

Contents

Figures

Overview

Objectives

The objective of the Robotics Developer Studio (RDS) Reference Platform Design is to provide guidance on the basic requirements for developing a "real" robot for use with services provided by RDS. A simulated version of the Reference Platform is also available with RDS so that developers can get started on developing robotics applications in a simulated environment without the need of robotic hardware.

There is a wide variety of robots available on the market with little to no clearly defined industry standards. Available robots range significantly in price and capabilities. The Reference Platform provides guidance on a minimum set of components and associated software services that will assist a user to begin using RDS. Developers can therefore concentrate more on writing robotics applications rather than the fundamentals of building a robot and getting it to work. It is also intended to help reduce the level of skill required and eases entry into the exciting and rapidly growing field of robotics.

The Reference Platform is primarily intended for education and research into personal robotics. There are many possible areas of investigation, which might include:

- Exploration and mapping;
- Human Robot Interaction; or
- Personal Services, e.g. information updates, media player, etc.

The purpose of this document is to provide guidance on the Reference Platform as well as to explain the rationale behind some of the design decisions. In some areas of the Reference Design the guidance deliberately non-specific; this method is intended to provide a workable solution while still allowing for variations in the implementation to accommodate cheaper or more functional solutions. While there are other areas of the Reference Design that implementations should adhere to rigidly in order to help guarantee compatibility.

Reference Platform Features

The Reference Platform Design takes advantage of the Microsoft Kinect™ sensor which not only provides 3D depth data but also has a built-in web camera. For this reason the Reference Platform Design is also referred to as MARK – **M**obile **A**utonomous **R**obot using **K**inect.

A typical MARK robot might look like the following low fidelity drawing:

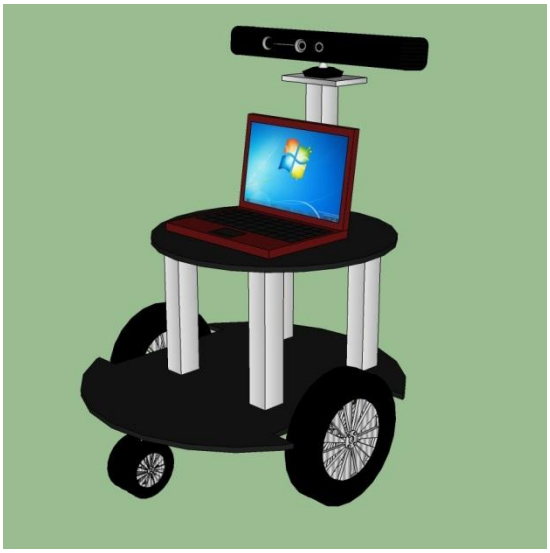

Figure 1 - Sketch of a Reference Platform

There are four key features inherent in a MARK design:

- Mobility
- Autonomy
- Interactivity
- Extensibility

Mobility is provided by a two-wheeled differential drive. This type of drive system is intended to be easy to control and allows the robot to turn on the spot. (Some designs might not turn within the robot's own diameter, but this feature is highly desirable to simplify navigation).

Autonomy is possible because the robot has an onboard PC. High-level functionality such as computer vision, speech recognition, network connectivity, etc. requires computers that are far more powerful than the simple microcontrollers used in low-end educational robots. Incorporating a PC into the design also allows wireless communication with other PCs and/or robots because WiFi interfaces are fairly ubiquitous on laptop and netbook computers.

Interactivity has two aspects:

- Human Robot Interaction (HRI); and
- Interaction with the environment.

The field of personal robotics has a need for easy to use and highly reliable User Interfaces. Interactivity is facilitated by the inclusion of a Kinect sensor which can dramatically simplify the coding of User Interfaces by providing 3D data and skeleton information. Interacting with the environment is also a key aspect of personal robots, such as navigating while avoiding obstacles, for which 3D data is also very useful.

Extensibilty is inherent in the Reference Design because it does not specify particular low-level hardware and is not restricted to a single solution. This allows Microsoft Robotics Partners and users alike to substitute different parts and vary the design so long as the major features remain intact. Microsoft Robotics Partners are encouraged to provide some built-in expandability, e.g. spare IO pins, extra power capacity, optional components, etc.

How to Interpret this Design Specification

Where an item is listed as "must have" it must be included in any robot kit that aims to qualify as an RDS Reference Platform and/or to be referred to as a MARK robot pursuant to the Microsoft Robotics Partner Program. Anything listed as a "basic requirement" is mandatory for use with RDS services.

Items that are listed as "recommended", "desirable", "should be" or "optional" are not essential and do not need to be included in a MARK robot. Vendors can add these or other items at their discretion.

Throughout this document various different prototype Reference Platforms are shown. These are not intended to be prescriptive, but rather to show that the design is flexible.

Vendors who are interested in selling MARK robots can become Microsoft Robotics Partners by contacting Microsoft at msrspp@microsoft.com and supplying the details of their kits.

Vendors who wish to sell MARK kits as a Microsoft Robotics Partner must cater for all of the items listed in the basic requirements and provide RDS services to interface with the hardware. Vendors can supply kits either assembled or unassembled. Regardless of whether the robots are supplied assembled or unassembled, vendors must supply assembly instructions showing all the steps required to build their robots from their constituent parts. Ideally there should be no requirement for tools other than common ones like a screwdriver, Allen key or a hammer to build the robot. Specifically, there should not be a requirement to do any soldering – all wiring connections should be screw type or quick connect (lug) type connectors.

Diagrams, CAD Drawings and a Bill of Materials (BOM) for Microsoft Robotics Partner MARK kits must also be supplied that show sufficient detail to enable an experienced robot builder to build a robot using off-the-shelf parts assuming that they have the requisite experience and access to the necessary tools. It is not expected that the average user will have a laser cutter or CNC machine, but it is expected that a user might build portions of the robot themselves using basic tools like a screwdriver, drill, or soldering iron rather than purchasing them from the Microsoft Robotics Partner. This is intended to allow users maximum flexibility and to make modifications to suit their own needs.

Individuals who wish to build a MARK robot can do so with no restrictions. In fact, independent developers are encouraged to build their own robots and write the supporting RDS services.

Hardware Design

This document provides a "blueprint" for the Reference Platform, or MARK. Anyone can build a robot based on this design. The design is not overly prescriptive, although there are certain key components that are required such as the Kinect sensor.

The Reference Platform Design is intended for indoor use only and should not be operated in environments with excessive moisture or extreme temperatures.

A fully built Reference Platform robot should look similar to the following pictures:

Figure 2 - Typical Reference Platform Robot – Front (left) and Back (right)

The components of a basic MARK kit need to include:

- Base and all mounting hardware
- Upper deck for mounting a PC
- Kinect mounting support and cable connection
- Motors and Wheels with Encoders
- Caster(s) for stability
- Battery and Charger
- Proximity sensors (combination of Infrared and Sonar)
- Voltage regulators (at least one 12V 2A regulator, or better)
- H-Bridges for Motor Control
- Robot IO Controller Board (for motors and sensors) capable of communicating with a PC

The following items might be included in the kit or be purchased separately to complete the robot as per the design. They are essential components of the design but are not a requirement for Microsoft Robotics Partner MARK kits.

- Laptop or Netbook computer (see Computer Requirements below)
- Kinect sensor
- Wireless Xbox Controller

There is no limit to the additional (optional) components that can be added to the robot provided that the corresponding DSS services are also available. In particular, Digital IO is not a required feature but it is highly desirable. Some examples of optional components are devices such as:

- Cliff Sensors
- Compass
- Accelerometer
- Gyroscope
- High resolution webcam (see note below on USB bandwidth)
- Digital IO pins and associated devices such as bump (contact) sensors, relays, etc.
- LEDs or lights for signaling or decoration

Main Mechanical Components

The basic design must have a minimum of two decks:

- A Base for the drive system and electronics; and
- An Upper Deck for mounting the PC and the Kinect.

The deck with the largest diameter must be round. (This will most likely be the Base). This is because round discs allow the robot to rotate on the spot without hitting anything.

All components mounted on the robot must fit within a virtual cylinder defined by extruding the largest deck upwards to the top of the robot and downwards to the floor. Put another way, none of the components should stick out beyond the decks. This includes the wheels. The reason for this requirement is to support turning in place. A robot that encounters a blocked path should always be able to rotate to find clear space.

The design does not limit the number of decks, so more can be added. The exact height and spacing of the decks are not specified either. For example, an additional deck might be desirable to cater for a variety of extra sensors.

The robot should be about the same width as a human so that it can navigate through doorways. This means that the diameter of the decks must typically be less than 50-60cm. (Most doors are 75cm or greater in width).

Base (Lower Deck)

The base provides support for the whole robot. It includes:

- Differential Drive (see below);
- Caster(s);
- Power System; and
- Sensors.

The base must be stable when turning at full speed or stopping from full speed.

Any MARK kit that is supplied by a Microsoft Robotics Partner as an unassembled set of parts must include all the necessary components including screws, nuts, bolts, mounting brackets, wiring, electronics, etc. Partial kits are not allowed (except that the computer, Kinect and Xbox Wireless Controller can be separate items).

Upper Deck

The upper deck is supported by uprights from the lower deck. It must be large enough to hold a typical laptop with a 15 inch (38 cm) screen and capable of holding the weight of a laptop. It does not have to be the same size as the Base, although it is not advisable to make it larger.

The material for the uprights is not specified. A good material for this purpose is 80/20 for example, although there are cheaper alternatives. The uprights must be strong enough to hold the upper deck, laptop and Kinect, and withstand occasional collisions, i.e. they need lateral as well as vertical strength.

Provision must be made for attaching a laptop to the upper deck. This might take the form of Velcro strips for instance. Regardless of the mechanism used, the necessary parts must be included in the MARK kit.

Ideally it should be possible to remove the upper deck for easy access to the base to work on the electronics mounted on it. This means that there should not be a lot of cabling between the Base and the upper deck, or the cables must be easily detachable or long enough that the upper deck can be laid aside once removed.

The upper deck also holds the Kinect Support as explained in the next section.

Kinect Support

The upper deck must have a support attached to it for the Kinect camera which must be strong enough to hold the weight of the Kinect and rigid so that the Kinect does not wave around when the robot is moving. (There might be some movement of the Kinect that results from the whole robot rocking backwards and forwards slightly due to the elasticity of the wheels and casters. Applications need to take account of this when they process the depth data. However, this movement is unavoidable and it must not result from flexing of the Kinect support.)

Once mounted, the height of the Kinect cameras should be around 60cm off the ground. Larger values are permissible, but the higher the Kinect is the less it will see of the ground in front of the robot when it is horizontal (parallel to the ground). If the Kinect is raised too high, it will need to be tilted downwards to be used for obstacle avoidance. In that case it will be unlikely to see people's faces. This is a compromise that is a consequence of the limited Field of View of the Kinect.

The Kinect must be attached to the support using clips or screws on the underside of its base. The Kinect must not require modification.

Attachment in this fashion leaves the Kinect free to tilt using the built-in servo mechanism.

NOTE: The Kinect is deliberately mounted towards the back of the upper deck. This leaves room for a laptop and it also helps to minimize the effect of the "blind zone" in the depth image. (See the Kinect section below for more details).

It is desirable to leave the laptop open so that the screen is visible while operating the robot. The Kinect Support might have an attachment to hold the Laptop screen in place and prevent it from waving backwards and forwards when the robot is moving.

Differential Drive

A differential drive consists of two driven wheels that can have their speed varied independently. Motor control is usually accomplished by using H-Bridges which are necessary to handle the high current required to drive the motors.

The drive system must include the motors, H-Bridges, wheels, axel couplings, and all mounting hardware (brackets/supports, nuts, bolts).

It is highly desirable that the robot be designed in such a way that it can be propped up off the ground so that its wheels to do touch the ground. This task should be easy to achieve and the result should be stable. For example, a block of wood or large book placed under the robot should raise it high enough that the wheels clear the ground but it does not rock backwards and forwards or easily fall off. The packaging materials for the robot might also be useful. This allows tests to be performed with the drive system enabled.

Motors

The motors can be of any type provided that they have sufficient power to drive the robot at normal walking speed (around 4-5 Kilometers per Hour). The maximum speed can be controlled via gearing or through software.

It is desirable that the motors be as quiet as possible during operation. Apart from noisy motors being annoying, too much noise makes it difficult or nearly impossible for speech recognition to work with the Kinect Microphone Array.

Wheels

Wheel diameter is not specified, but it is recommended that the wheels be approximately 15cm in diameter.

It is recommended that all the wheels (main drive wheels and casters) have some compliance, i.e. they should have some elasticity rather than being rigid.

Because a differential drive consists of only two wheels, a third caster or jockey wheel is required for stability. However, a three-wheeled design might result in the drive wheels being off-center so that the robot cannot rotate within its own diameter. Adding a second caster (one in the front and one at the back) provides a stable configuration and ensures that the robot can rotate within its own diameter. The following photograph shows the underside of a prototype robot.

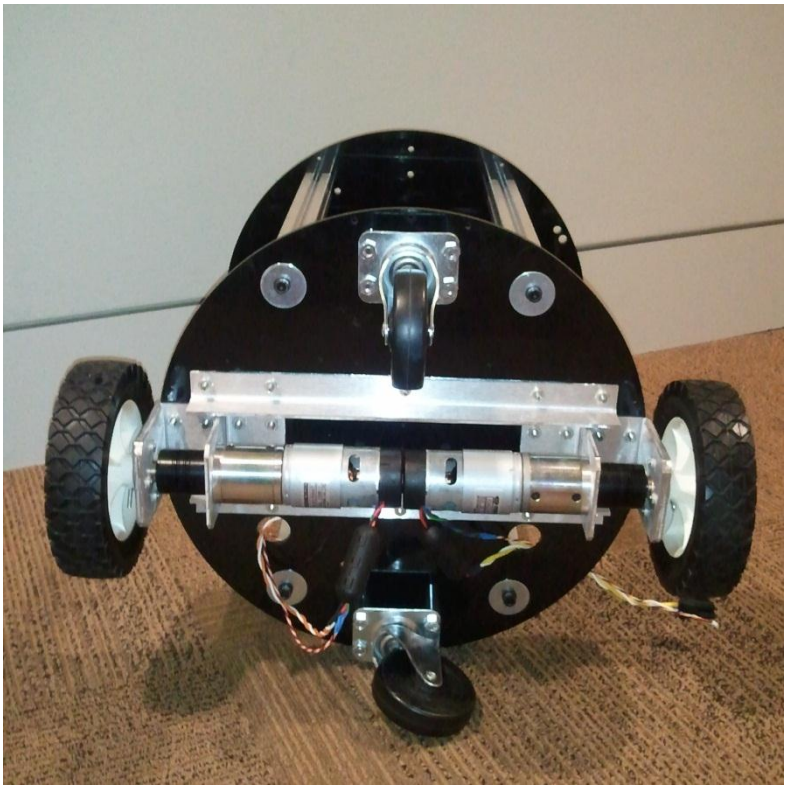

Figure 3 - Wheel Geometry – Underside of a robot

In this particular design, the wheels extend outside the base. The robot width (diameter) must be specified at the widest point, which in this case is the width between the outside of the wheels. The robot width is important for navigation because it determines the size of the spaces that the robot can move into.

Ground clearance is not specified because the robot design is intended to be used indoors on flat surfaces with no small obstacles. However, even when used indoors, there might sometimes be transitions between flooring types, e.g. carpet to hardwood, which the robot might not be able to negotiate. Any obstacle higher than a couple of centimeters can cause a four-wheeled design to strand itself when the front caster rides up onto the obstacle and the drive wheels lose traction.

Encoders

The Wheel Encoders must provide feedback at a frequency of 20Hz or better with a resolution of two degrees of wheel rotation or better, i.e. 180 "ticks" (or counts) per revolution.

Power System

All Microsoft Robotic Partners producing MARK Kits should have at the very least a good understanding of basic electrical safety and an awareness of the potential hazards associated with working with electrical products. Moreover, they should be knowledgeable about ways to reduce and eliminate potential hazards such as use of insulation, guarding, electrical protection devices, and safe work practices.

The power system must provide power for all of the robot components, except for the laptop which is optional. This includes power for the following devices:

- Kinect (regulated 12V at 1.1A)
- Motors and motor controllers
- Robot IO Controller
- Proximity Sensors

Additional regulated power (5V, 3.3V) is optional.

It is desirable that there are lights or LEDs on the robot that are illuminated when the power is on, but there is no requirement for a separate power light if other components have lights.

An indication of low battery level is also desirable. SLA batteries will survive a heavy discharge (although it is not good for them) but this might result in erratic behavior if the main voltage regulator cannot maintain 12V with a very low input voltage. It is recommended that the battery voltage be read via an analog input so that applications can respond to low battery situations.

The following photograph shows the key components of the power system including a voltage regulator, power rails, power switches and charging connector. (Lights are an optional extra).

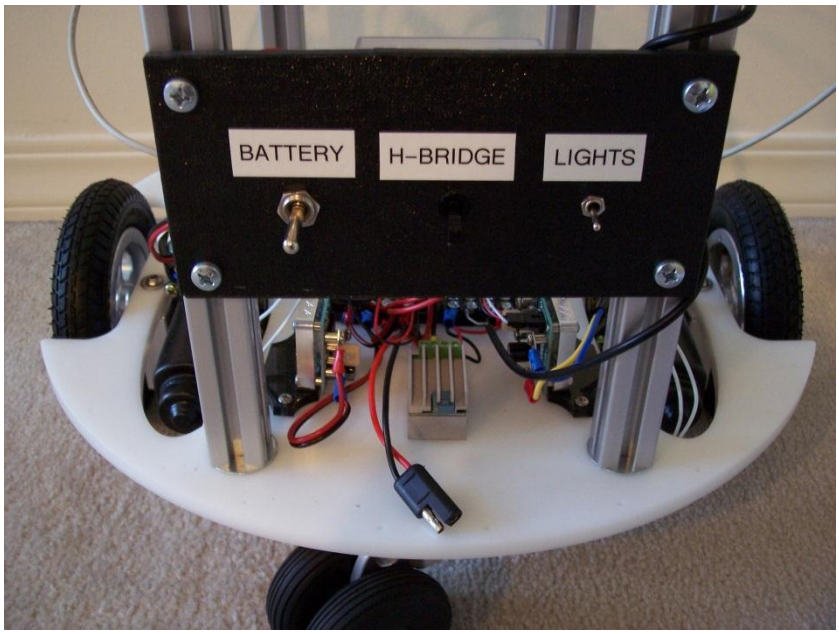

Figure 4 - Key Power System Components (Rear view)

Switches and Wiring

There should be separate switches for main battery power and the power to the motors (or H-Bridges) in an easily accessible location. This allows the motors to be disabled independently of the main power.

All switches must be clearly labeled. They must be rated to handle the maximum expected voltage and current.

The following photograph shows another possible implementation of the power switches. Note that it is different from the solution in the previous photograph.

Figure 5 - Power Switches (Rear view)

It is also desirable to have a Quick Stop switch that is readily accessible. This can be used to stop the robot in the event of a software or hardware malfunction.

All necessary power wiring must be supplied in the appropriate lengths and with the necessary connectors already fitted (crimped or soldered on). Connectors must be appropriately rated and wiring must be sized to handle the maximum possible current. Fuse(s) should be included in the power circuit(s).

Power rails must include at least one spare connection for 12V power. If additional voltage regulators are included, then a spare connection should be available for each voltage level.

Batteries

The power for the robot must be supplied by one or more batteries. The battery chemistry is not specified. Experience has shown that 12 Volt Sealed Lead Acid (SLA) batteries provide sufficient battery life and can be relatively easy to recharge.

The battery capacity must be sufficient to enable the robot to operate for a minimum of 2 hours between recharges assuming that the motors are only used for 1 hour. There is no requirement for the computer to run off battery power if it is a laptop or netbook, although provision for this is desirable. In any case, running the computer off the robot's battery power must not reduce the operating time below 2 hours.

For classroom use, it is desirable for the battery life to be 5 to 6 hours. For example, 14Ah capacity will most likely satisfy this requirement. This will allow the robot to be used throughout a normal day by multiple students with their own laptops and then recharged overnight. (This battery life exceeds that of all but the best laptops. However, using multiple laptops removes the laptop battery life as the limiting factor.)

It is recommended that the batteries be mounted below the base to keep the center of gravity low (as in the photograph below). This does have the undesirable side-effect of reducing the ground clearance. However, the Reference Platform is designed for indoor use and it is not expected to

navigate over obstacles of any significant size. The batteries must be securely mounted so that they do not move around in normal use. Consideration should also be given to ease of battery replacement.

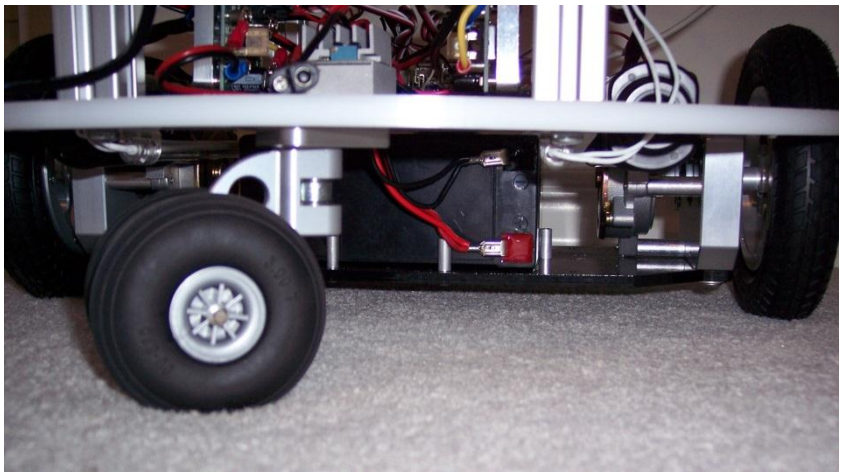

Figure 6 - Battery Mounting under the robot

Charger

A charger must be supplied with the robot and it must be capable of recharging the batteries in 4 hours or less. The connection between the charger and the robot must be by means of a keyed connector, i.e. it must be impossible to invert the polarity when plugging in the charger.

Chargers supplied with a robot must include safety features such as battery overcharging protection, short-circuit protection, over-current protection and reverse polarity protection. The AC adaptor and wiring must be suitably rated for the capacity of the battery; able to safely handle the current required to charge the battery; and be properly grounded and/or insulated.

Operation of the robot while it is attached to the charger is desirable, but not required. There is a risk of damage to the robot and/or charger if the robot drives away while tethered. This is one of the reasons for a separate motor switch.

Proximity Sensors

The Kinect depth data can be used for obstacle avoidance. However, the Kinect has limitations (as explained in the next section). Therefore separate proximity sensors, or obstacle avoidance sensors, are required.

The proximity sensors must consist of a combination of at least two distinct types of sensors. Infrared and Sonar sensors are recommended. The reason for this requirement is that different types of sensors have different failure modes, so combining different types increases the likelihood of detecting obstacles. For instance, IR sensors see straight through glass. Sonar sensors however detect glass surfaces very well. Conversely, sonar (ultrasound) signals bounce off surfaces that are at a small angle of incidence to the sonar beam, so there might not be enough signal returned (the "ping") to be measured. IR sensors use back-scatter of the IR light and can still work at low angles of incidence.

The recommended combination is three IR sensors and two sonar sensors. These must adequately cover the area in front of the robot from approximately 10cm out to 60cm from the front-most point of the robot. The photograph below shows a robot with the suggested sensor arrangement.

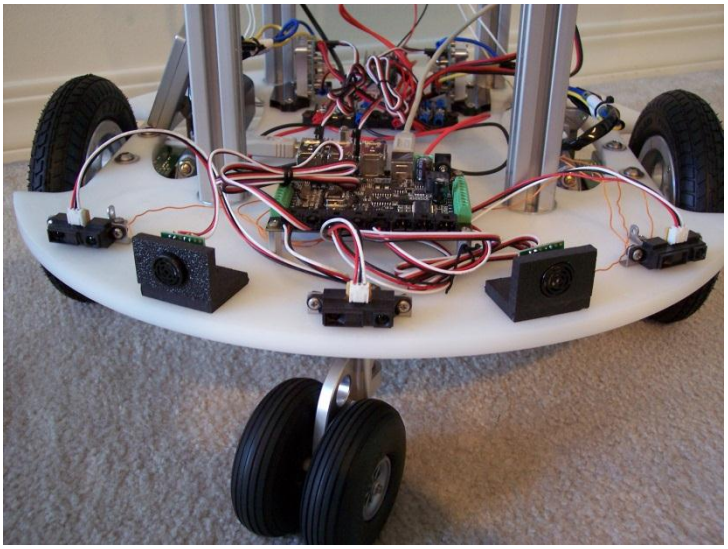

Figure 7 - Proximity Sensors

All necessary mounting hardware (brackets, screws, spacers, etc.) and cables must be included in the kit.

There must be an Analog Sensor Array service provided that exposes the measurements from these sensors with a frequency of at least 10Hz. The units and range of the various sensors must be documented. Ideally the service should convert all measurements to centimeters. The Pose information (Position and Orientation) of the sensors must be included in the sensor definitions for the sensor service(s). If the Poses are adjustable by the user, there must be clear documentation on how to change the software configuration.

There is no requirement to have side-facing or rear-facing proximity sensors, although this is desirable. Side-facing sensors are useful for wall following. Rear facing sensors are necessary when backing up because the Kinect cannot turn around. However, it is recommended that the robot should not be driven backwards. Instead it should always rotate on the spot and then drive forwards.

There is no requirement for the sensors to all be located on the lower deck, but for avoiding obstacles it makes sense to keep the sensors low. Alternatively, they could be mounted to the bottom of the upper deck. The sensors might need to be tilted up slightly to avoid seeing the floor as a constant obstacle.

Robot IO Controller

The Robot IO Controller, also called the Brick, is a circuit board that acts as the interface between the robot hardware and the PC. Typically the Brick will be a single board, but this is not required as long as it logically appears as a single board to higher-level software services.

The following photograph shows an implementation that uses two separate controller boards. (The top deck is transparent enabling a top-down view of the base).

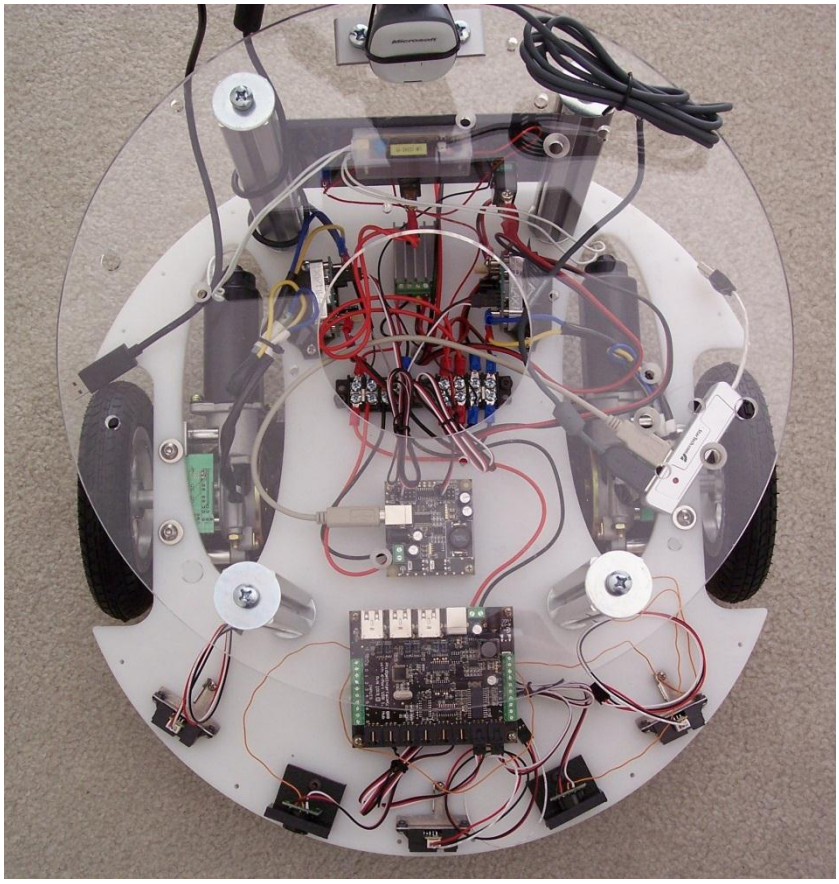

Figure 8 - Robot IO Controller and Associated Electronics

NOTE: To minimize clutter in the photograph, there is no USB cable connected to the Analog/Digital IO board. The sensor cables are not connected either. The USB cable plugs into the small unpowered USB hub at the right-hand side near the wheel. Care must be taken to avoid overloading the USB port on the PC with too many devices. This particular configuration is close to the threshold for power supplied by a single USB port (500mA).

A Brick service must be supplied that is able to control the motors and read the encoders and proximity sensors.

In the basic design the Brick must use USB for communication with the PC (as shown in the photograph above).

NOTE: See the section on USB Constraints which also applies to the Brick.

Brick designs that use a serial port must include a USB-to-Serial adapter either on the board itself or as a separate dongle that is included in the kit. In other words, the interface to the PC must still be USB.

An alternative approach to connecting the Brick is to use Ethernet. This is allowable if the Brick has an on-board Ethernet port or an adapter that is included in the kit.

All necessary cabling or mounting hardware, e.g. screws and standoffs, for the Brick must be included in the kit.

Any Firmware required to run in the Robot IO Controller must be supplied by the Microsoft Robotics Partner or written by the user. RDS cannot be used to generate Firmware for non-Windows platforms. Visual Studio can be used to compile code for boards that implement the .NET Micro Framework, but this is a separate package from RDS.

Kinect

Any Microsoft Robotics Partner MARK kits must provide a solution that does not require the user to modify the Kinect physically or electrically. This means that the Vendor must supply the appropriate connector for plugging the Kinect into the robot.

Microsoft Robotics Partners should consider providing a small quantity of cable ties in the kit to help to wrap up the Kinect cable and other cables such as the Xbox Wireless Controller.

Specifications

The Kinect has a 57 degree horizontal Field of View (FOV) and 43 degree vertical FOV. The highest depth resolution is 640x480 pixels. Several lower resolutions are also supported.

All range values from the Kinect are measured in millimeters. These values are measured perpendicular to the major axis of the Kinect, i.e. they are not radial values. The values are quantized in a non-linear way due to perspective effects, i.e. the potential error in a measurement increases as the range increases.

The Kinect depth camera cannot "see" below 80cm (800mm). This is a limitation that results from the requirement to separate the laser from the camera by a precisely known baseline distance. In effect, the camera becomes "cross-eyed" at smaller ranges. (The region below 80cm is referred to in this document as the "blind zone"). Therefore the camera cannot see objects immediately in front of the robot because the robot diameter must be less than 60cm.

The maximum depth range of the camera is restricted to 4m (4000mm) which is the maximum range that the Kinect was originally designed for.

Theory of Operation

The Kinect depth camera consists of two parts:

- Infrared Laser; and
- Infrared camera.

In the photograph below, the window on the left is the IR laser. The one on the right is the IR camera. The one in the middle is an RGB camera.

Figure 9 - Kinect Sensor

The Kinect RGB camera is a conventional webcam. It is located close to the Kinect IR camera so that the depth and RGB images align as closely as possible, but they cannot be coincident so there are always some differences between the images.

The IR laser sends out a pseudo-random dot pattern and the reflected dots are read by the IR camera. Objects in the Kinect's FOV distort the dot pattern. (A perfectly flat surface perpendicular to the Kinect would result in an exact match to the original dot pattern arrangement, but the spacing between the dots would increase as the surface was moved away from the Kinect). The laser and camera must be separated by a precisely known distance for depth measurements to work. A processor inside the Kinect matches the reflected dots against the known pattern and calculates the distance based on the offset of each dot from its expected position by using a method similar to triangulation.

The following photograph, taken using a camera that detects infrared, shows the spots generated by a Kinect camera.

Figure 10 - Kinect Infrared Dot Pattern

Notice that there is a distinct shadow behind the person in the photo. Because the laser and the camera are separated by some distance, they have slightly different viewpoints. This means that shadows or "halos" appear around objects where the Kinect cannot determine the depth. These occlusions result in pixels with a "no reading" value (zero) in the depth image.

Also visible in the photo are the boundaries of the Kinect's FOV across the ceiling and the wall at the right-hand side.

Computer Requirements

Various different netbook/laptop computers have been tested with a Reference Platform implementation. This section outlines the computer requirements and gives some guidance on choosing an appropriate computer.

The preferred computer is a laptop or netbook because many people already have one. It can also simplify software development because the same computer can be used for both development and running the robot. In a classroom situation, one robot can easily be shared amongst multiple students with their own laptops because only USB connections are required to connect to the robot making it easy to attach and detach a computer.

Minimum Requirements

The PC must be capable of running Windows 7 and RDS, including:

- DirectX 9.0c compatible graphics card
- Dual-core processor (2GHz or faster recommended)
- 10GB of available disk space
- 2GB of memory (4GB recommended)
- At least 2 separate USB 2.0 channels (ideally 3 or more)

Optional Requirements

- WiFi adapter

NOTE: Almost all laptop and netbook computers manufactured over the last few years have WiFi. It is not part of the basic requirements because the robot is intended to operate autonomously.

The weight of the computer should not be a significant issue because the Reference Platform should be powerful enough to carry the weight. However, it is necessary to attach the computer to the robot so that it does not fall off and this can be complicated depending on the size/shape of the computer. Computers with screen sizes of more than 15 inches (38cm) might be too large to fit on the robot's top deck without extending out beyond the edges of the deck.

Depending on the purpose of the robot, the exact requirements for the PC will vary. As a rough guide, one way to test the computer performance is to install the Kinect for Windows SDK and run the Skeleton Viewer. The following table shows figures that are representative of different computers but they are not the result of stringent testing.

Computer	CPU	Frame Rate
Asus 1215N Netbook	Intel Atom D525 1.8GHz	5 – 10 fps
Asus Eee Slate	Intel Core i5 1.33GHz	15 – 20 fps
Dell Inspiron 15R	Intel Core i3 2.53GHz	25 – 30 fps
Lenovo T61p	Intel Core2 Duo 2.2GHz	20 – 30 fps
Lenovo IdeaPad Z370	Intel Core i5 2.3GHz	25 – 30 fps

NOTE: Microsoft does not recommend any particular brand or model of computer. The choice is up to the Microsoft Robotics Partner or the end user. The examples above are indicative only and do not constitute any form of benchmarking.

Using an ATX motherboard on the robot (instead of a laptop or netbook) will most likely require the addition of another voltage regulator because it will require more current than can be supplied with the specified on-board regulators. A motherboard solution will most likely not include a display screen. While this is acceptable for some applications, it could be a limitation for HRI if visual feedback is required. Therefore the basic design specifies a laptop or netbook.

USB Constraints

When selecting a computer to use with the Reference Platform, a variety of factors need to be considered such as CPU power, memory and disk storage capacity. Factors that are less obvious are USB bandwidth and power consumption.

The Kinect sensor uses a large amount of bandwidth, particularly when reporting both depth and RGB data simultaneously. This could have adverse effects on other devices attached to the same USB

controller. It is therefore recommended that the Kinect be attached to a separate USB 2.0 (or 3.0) channel. Some low-end computers on the market only have one USB controller. Test a computer with the Reference Platform to confirm that USB bandwidth is not an issue.

It is recommended that any additional devices be placed on a separate USB controller from the Kinect.

Adding a webcam to the robot also has an impact on the USB controller. Webcams that request more than 50% of the USB bandwidth will either not start or will prevent the Kinect from starting.

Another consideration is that the Xbox Wireless Receiver uses a lot of USB power. A USB channel has a maximum power output of 500mA. Attaching several devices to an unpowered hub, including the Xbox Wireless Receiver, might cause erratic behavior. For instance, the Xbox Controller might not pair with the Wireless Receiver, or another USB device might fail to operate as expected.

Do not connect the Kinect to the same unpowered hub as an Xbox Wireless Receiver. This combination is known to be incompatible.

NOTE: A powered hub can alleviate the power problem of the Xbox Wireless Receiver. However, if a powered hub is included in the design then it must be powered by the batteries.

Software Architecture

Overview

The basic software architecture is shown in the diagram below:

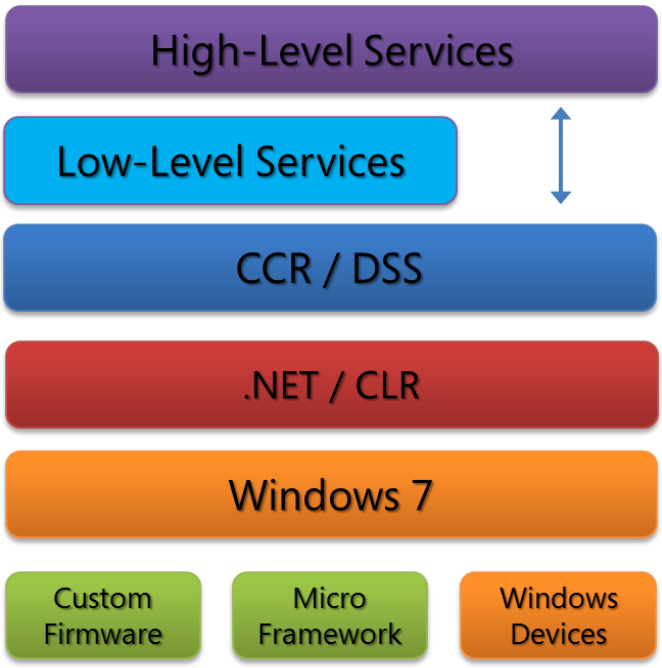

Figure 11 - Software Architecture

Various High-Level Services are supplied as part of RDS, or can be written by third-parties. These services typically rely on abstractions provided by Low-Level Services to communicate with devices. However, some High-Level Services operate on their own.

The Low-Level Services must be provided by the maker of a MARK robot or written by a user if they are building their own robot. These services communicate with the Robot IO Controller. They are similar to a Hardware Abstraction Layer and present defined interfaces (referred to as Contracts in RDS) to the High-Level Services.

Services for the Kinect and several other devices are included with RDS.

CCR (Concurrency and Coordination Runtime) and DSS (Decentralized Software Services) are components of Robotics Developer Studio. All services run within the context of a DSS Node, and multiple DSS Nodes can communicate with each other over a network.

RDS is based on .NET, which is a pre-requisite, and this in turn runs on Windows.

At the bottom level is the hardware. Some devices, such as the Wireless Xbox Controller, are directly supported by Windows drivers. The Robot IO Controller board contains Firmware that communicates via Windows using a Serial Port, USB, network connection, or some other means.

Taking a different view, the following diagram shows how a combination of specific High-Level and Low-Level services can be used to communicate with the hardware on the robot. These services are

"orchestrated" by means of manifests that list the required services and the partnerships amongst them (represented by arrows).

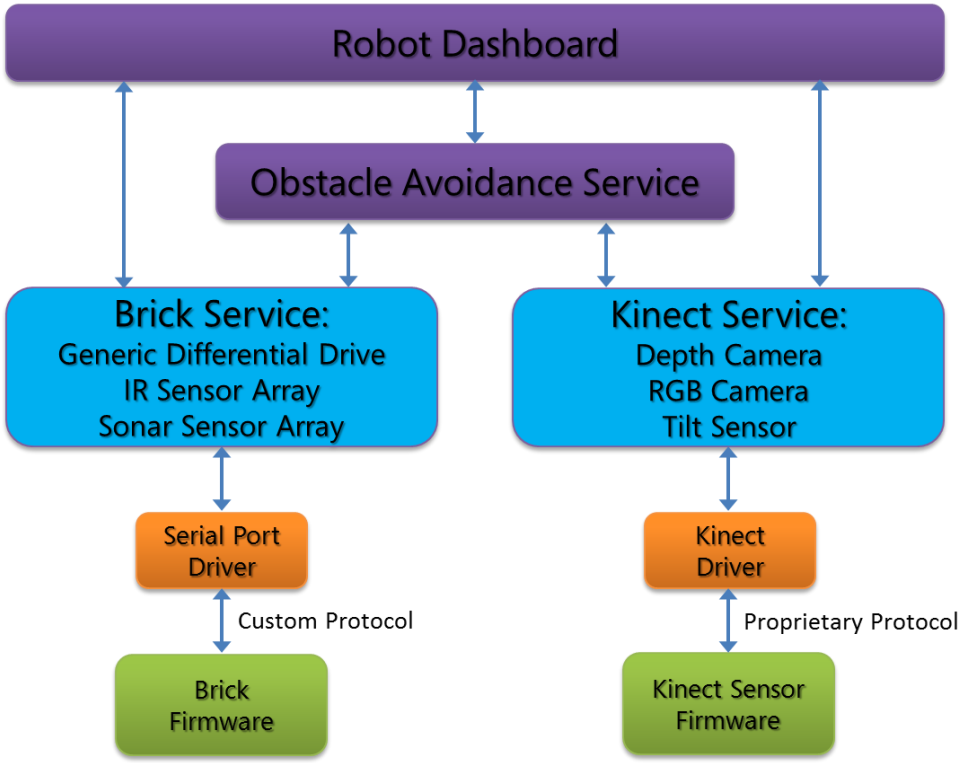

Figure 12 - Orchestration of DSS Services

Starting from the top, the Robot Dashboard is a High-Level service that allows users to drive the robot and view status information. It is independent of the particular type or model of robot, as long as the robot conforms to this specification and the appropriate Low-Level services are provided. The Robot Dashboard is included in the RDS package.

In between the Robot Dashboard and the Low-Level services is another High-Level service that implements obstacle avoidance. From the point of view of the Robot Dashboard, it appears like any other Generic Differential Drive because it exposes the same contract. By simply changing the manifest this layer can be removed and the Robot Dashboard can talk directly to the Generic Differential Drive, i.e. direct drive. (The Robot Dashboard could in fact switch between the two drive services, but this is an unnecessary complication at this level of explanation).

The Brick Service has its own "native" *contract* (interface) that can expose specific features of the robot that are outside the scope of this specification. However, the primary purpose of the Brick Service is to provide contracts for:

- Generic Differential Drive
- Wheel Encoders
- IR Sensor Array
- Sonar Sensor Array

(Additional services might be required in future versions of this specification).

The implementation of the Brick is open to the developer. It might consist of a set of separate services or it could be implemented as a single service that offers *alternate contracts*.

These services alone are sufficient for rudimentary navigation and obstacle avoidance. However, the Kinect adds an entirely new dimension to obstacle avoidance by providing 3D depth data. The Kinect service implements the following contracts:

- Kinect Sensor
- Depth Cam Sensor
- Web Cam Sensor
- Skeleton Tracking
- Microphone Array

The Kinect Sensor exposes the Tilt mechanism. The purpose of the Depth Cam and Web Cam sensors is obvious. Skeleton Tracking and the Microphone Array are not available in the Simulated Kinect. However, RDS provides a single microphone input as a standard service.

The Low-Level services talk directly to Windows drivers – in a sense they are "wrappers" for these drivers to incorporate them into the CCR/DSS environment.

At the bottom of the stack are the Firmware implementations for the various hardware components.

Pre-Requisites

To use a Reference Platform with RDS the following software must be installed in the order specified:

- Windows 7 (32-bit or 64-bit)
- Visual Studio 2010 Express or higher
- Kinect for Windows SDK Beta (and associated pre-requisites)
- Robotics Developer Studio 4 (and associated pre-requisites)

Because RDS includes a sophisticated simulator, it is possible to verify the installation and even begin working on robotics applications without a real robot.

The following sections contain more details about the pre-requisites.

Visual Studio 2010

Visual C# Express is available for download at:
http://www.microsoft.com/visualstudio/en-us/products/2010-editions/visual-csharp-express

NOTE: It is recommended that Professional Edition or above be used with RDS. The Express Edition has limitations that can affect usage of RDS in some situations.

Students may be able to obtain Visual Studio through the DreamSpark program from:
https://www.dreamspark.com/.

The latest updates and service packs should be applied.

Kinect for Windows SDK Beta

The Microsoft Kinect for Windows SDK Beta is available for download from Microsoft Research at:
http://research.microsoft.com/kinectsdk

IMPORTANT NOTE: Before using a new Kinect attached to a PC, plug it into an Xbox that has network access so that the Xbox can update the firmware in the Kinect. Otherwise it might not work reliably.

The Kinect for Windows SDK requires DirectX SDK, Microsoft Speech Platform and the Kinect for Windows Language Pack (for acoustic models).

At the time this documentation was written, the Kinect for Windows SDK could only be used with 32-bit applications. Therefore all applications must target a 32-bit environment, not a 64-bit environment.

Robotics Developer Studio

Robotics Developer Studio 4 (or later) is required to support the Reference Platform. RDS is supplied as a self-installing executable. An Administrator login is required to install RDS.

RDS requires .NET 4.0 (installed with Visual Studio), DirectX 9.0c, XNA 4.0 and PhysX 2.8.1. The pre-requisites are installed automatically except for .NET.

Required Services

For a robot to be classified as a MARK robot pursuant to the Microsoft Robotics Partner Program it must have the necessary RDS services available. These services include:

- Drive service that implements the Generic Differential Drive contract;
- Encoder service that implements the Generic Encoder contract; and
- Analog Sensor services that implement the Analog Sensor Array contract for the IR and Sonar sensors.

It is optional to provide a GPIO service for handling Digital IO.

Brick Service

It is feasible to create a single Brick service that implements all of these contracts as alternate contracts, i.e. the Brick service appears logically as multiple services. There is currently no requirement for a Brick service in its own right.

NOTE: Future versions of this specification might include details of a Brick service.

Regardless of how the services are implemented, the software package must also include the necessary configuration files and manifests to run the services. The configuration information must include the physical geometry details (Pose) of the drive as per the Generic Differential Drive contract and the details of the analog sensors as per the Analog Sensor contract.

Drive Service

There is no requirement for separate Motor services for each of the wheels. The implementation of the Drive service is free to implement the Motors internally and just implement the Differential Drive operations.

Ideally the Drive service should implement a PID control loop on the wheel rotational velocities, but this is not a mandatory requirement. (This function could also be the responsibility of the Brick Firmware). Wheel encoders are essential for PID control.

The Drive service must implement the **DriveDistance** and **RotateDegrees** operations with an accuracy of ±10%. At this level of accuracy, this is not a rigorous requirement. However, it provides developers with the basic capability to perform approximate right-angle turns, e.g. to go through a doorway, or to approach people and objects with reasonable certainty.

Sensor Services

Encoders must be provided. Due to the latency of passing messages up and down the software stack and the communications overhead with the Brick, it might not be feasible to use encoder values for precise control of the robot. This will vary from robot to robot.

Analog sensors of various kinds can be implemented. The minimum requirement is proximity sensors on the front of the robot.

It is desirable to provide some indication of battery level. This might take the form of a voltage reading presented as an analog value via the Analog Sensor Array. It is not required to implement the Generic Battery Contract.

Optional Services

Microsoft Robotics Partners who supply optional components must also supply appropriate RDS services for them.

High-Level Services

The following High-Level services are provided by RDS:

Service	Availability	Notes
Obstacle Avoidance Drive	RDS package	A variant of the Differential Drive that implements obstacle avoidance
Robot Dashboard	RDS package	A User Interface for controlling the robot and viewing output from the Kinect
Teleoperation	RDS / Windows	Not a separate service – Use Robot Dashboard

Microsoft Robotics Partners are encouraged to provide additional services to add value to their MARK kits. Likewise, members of the RDS community are encouraged to share their services.

Robot Dashboard

The Robot Dashboard provides a way to drive the robot remotely using a Wireless Xbox Controller. It also allows the operation of the various components to be verified: Kinect sensor; Drive Controller; and Proximity sensors. It works with both real and simulated Reference Platforms.

The following composite screenshot shows all of the windows that make up the Robot Dashboard.

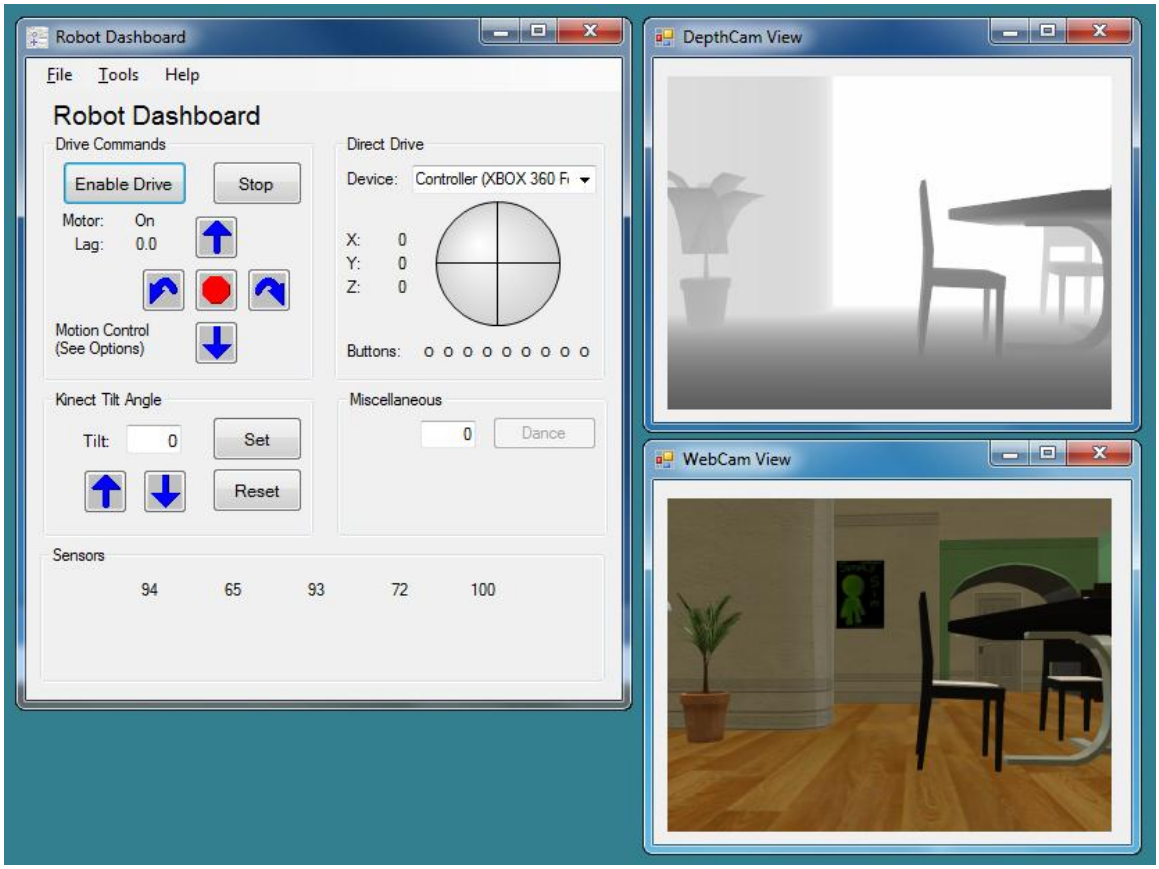

Figure 13 - Robot Dashboard

Buttons in the Robot Dashboard allow testing of the **DriveDistance** and **RotateDegrees** functions, as well as tilting the Kinect camera. A Wireless Xbox Controller can be used to control the robot as well as the camera tilt.

The large trackball on the right is intended for use via Teleoperation. It is not feasible to use a mouse or touchpad on the robot unless it is up on blocks so that it does not move.

Sensor values are display across the bottom of the window to allow the sensors to be tested.

NOTE: The User Interface is subject to change between the Beta and the final version of RDS 4.

The Depth data and RGB (webcam) images are displayed in separate widows that can be independently resized and moved. This allows rearrangement of the interface to suit the user.

Notice in the DepthCam View window that the background disappears into white. This indicates that parts of the scene have exceeded the maximum range of the Depth camera.

Obstacle Avoidance Drive

The Obstacle Avoidance Drive service is designed to simplify writing applications for a MARK robot. It performs sensor fusion on the IR, Sonar and Depth data to detect obstacles and then adjusts the robots wheel speeds accordingly.

The service exposes a Generic Differential Drive contract so that other services can use it in the same way as any other drive service. The Obstacle Avoidance Drive service must partner with a real drive service to perform its job. This partnership can be specified in the manifest.

IR and Sonar sensors are optional partners for the Obstacle Avoidance Drive service, i.e. it will run even if these other services are not present (or not wired up in the manifest). However, it does require a Kinect Depth Cam service in order to run.

When operating only on Depth data, the Obstacle Avoidance Drive service cannot detect objects such as glass doors. Therefore it is highly advisable to add IR and Sonar sensors.

There is no User Interface for the Obstacle Avoidance Drive service.

Teleoperation

Teleloperation is the remote operation of a robot. For the Reference Platform it can be accomplished by two different methods:

- Use Terminal Services to connect to the PC over WiFi and then run the Robot Dashboard.
- Run the Robot Dashboard on another PC and specify that the robot services reside on a different computer. This requires a custom manifest and possible changes to the security settings for DSS.

NOTE: In the RDS 4 Beta it might not be possible to remotely view the Kinect data. This will be corrected for the final release.

Service Availability

The following table shows the availability of RDS services to support the Reference Platform. Additional capabilities might be added in future releases of RDS.

Service	Availability	Notes
Brick (Robot IO Controller)	Supplied by the robot maker	Must include the drive, encoder and analog sensor services.
Generic Differential Drive	RDS package	Contract Definition only – not used directly
Analog Sensor Array	RDS package	Contract Definition only
Serial Port	RDS package	Provides communication via a Serial Port (or virtual Serial Port over USB)
Kinect	RDS package	Depth, RGB, Tilt, Microphone Array and Skeleton Tracking services
Text To Speech	RDS package	Speech recognition using either the Kinect Microphone Array or a single microphone
Xbox Controller	RDS package	Wired or Wireless Xbox Controller
Webcam	RDS package	Can be used with any camera supported by Windows as a Webcam

Visual Simulation Environment

Getting started with the Visual Simulation Environment can be easy because a sample is included that implements a simulated Reference Platform robot which includes a Kinect sensor. The following screenshot shows a robot in the Simulation Editor.

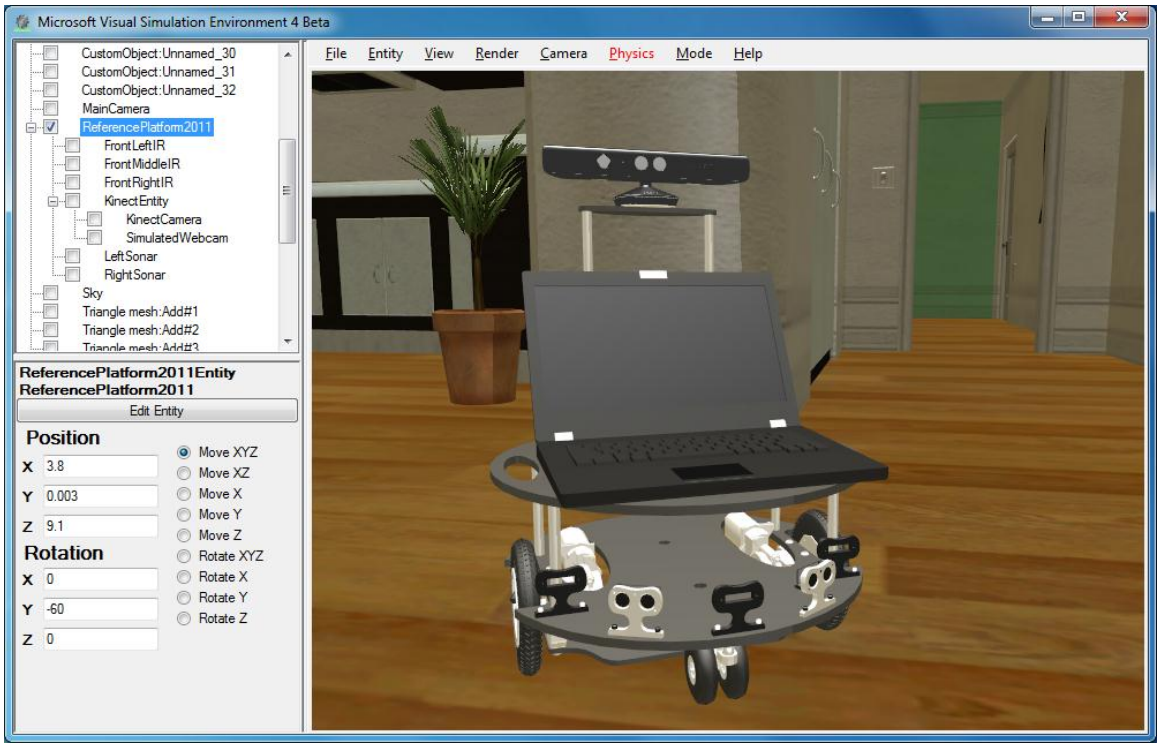

Figure 14 - Simulated Reference Platform Robot with Kinect Sensor

The simulated Kinect supports Depth, RGB and Tilt. The robot also has IR and Sonar sensors and a Differential Drive.

NOTE: In the RDS 4 Beta release the sonar sensors are implemented as small laser range finders. These sensors are likely to change in the final release.

Optional Components

There are several devices that are not included in the basic design. You can add your own devices if you write the necessary services or you can find suitable services on the Internet, e.g. on CodePlex.

Bump Sensors

Bump sensors could be attached to the robot in a variety of different ways and at different locations around its periphery. These are left to the ingenuity of the robot designer. They will require digital inputs on the Robot IO Controller.

Cliff Sensors

The Reference Platform Design does not include cliff sensors, i.e. downward facing distance sensors that can detect a drop-off, because it is intended to be used in controlled indoor environments on flat surfaces. Users of robots without cliff sensors should be careful when operating the robots near sharp declines, such as stairs.

Cliff sensors could easily be added to a robot if it has spare analog inputs.

Compass

Electronic compasses can be relatively inexpensive. They can be affected by large metal objects such as filing cabinets, refrigerators, or support beams in multi-story buildings so that the compass does not always point to magnetic north. However, the reading from a compass should be reasonably stable for any indoor location even if it is not correct in an absolute sense.

A compass can be a useful device for disambiguating the robot's heading. For instance, when travelling in a long office corridor the view in either direction might be very similar. Even a low resolution compass would allow the robot to distinguish one direction from the other.

Accelerometer / Gyroscope

There are various types of accelerometers and gyroscopes available. The price increases with the number of axes (1, 2 or 3). A 3-axis accelerometer plus gyroscope is likely to be more expensive than a single-axis accelerometer because the 3-axis accelerometer requires more complex interfaces than simple analog inputs.

Processing data from these devices requires a high sampling rate because the data often includes brief transient spikes. The average values can be used to verify that the robot is moving (and is not stuck) as well as the orientation of the robot (by looking at the gravity vector).

One alternative to an accelerometer is an inclinometer. This can be used to detect if the robot falls over. However, the four-wheel design reduces the likelihood of tipping.

Other Sensor Types

There are many other types of sensors that might be added to the robot for specific purposes, e.g. Temperature, Humidity, Light Level, etc.

Lights

Adding lights to the robot can make it more fun for students and attracts attention at trade shows. The following photograph shows a robot with the lights turned on. A separate power switch is recommended to avoid running down the battery unnecessarily.

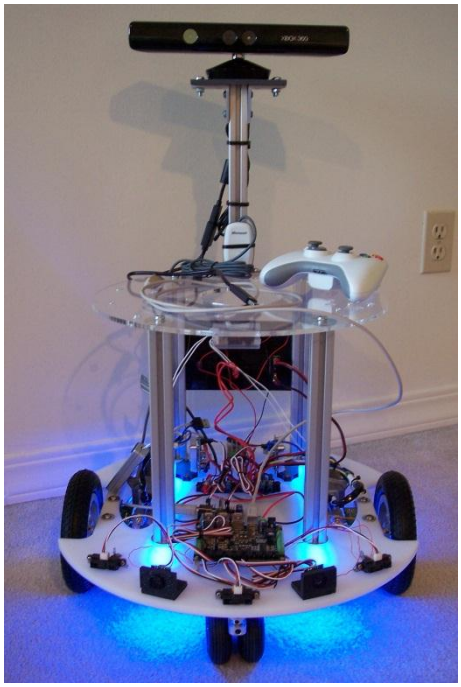

Figure 15 - Robot with Lights

The addition of LED indicators might also be useful for showing the status of the robot such as the power level or error conditions.

Further Information

For more information on RDS, visit the Robotics web site:

http://www.microsoft.com/robotics

The RDS documentation is available online in the MSDN Library at:

http://msdn.microsoft.com/en-us/robotics/default.aspx - MSDN Home Page

http://msdn.microsoft.com/library/bb881626.aspx - User's Guide and Class Reference (APIs)

For technical support, post questions to one of the Robotics Discussion Forums:

http://social.msdn.microsoft.com/Forums/en-US/category/robotics

The RDS software can be downloaded from:

http://msdn.microsoft.com/en-us/robotics/aa731520

Community-supported software is available from a number of sites on the Internet. In particular, search for "Robotics Studio" on CodePlex. The main CodePlex page for RDS is:

http://mrdssamples.codeplex.com/