Carnegie Mellon Robotics Academy

SMART Robotics Technician KSA

Knowledge, Skills, and Attitudes list

About this List

This list is the result of a research project funded by the ARM Institute, called the Smart Manufacturing and Advanced Robotics Training (SMART) Project. SMART consists of five micro-certification courses that focus on the Knowledge, Skills, and Attitudes (KSA) that are needed for Robotics Technicians in the Advanced Manufacturing and Robotics industry. The KSA was identified through research which involved interviewing and observing Robotics Technicians in the industry to ask questions like:

- What skills do you need to be successful in your position?
- What tools do you use most frequently in your work?
- What does your day-to-day look like?
- What kind of teammate would you like to have working alongside you?

Hiring managers were also interviewed to understand what type of workers they look for to join their company. The SMART program consists of five micro-certification courses (estimated 40 hours each, 200 hours total):

1. Mechanical Foundations
2. Electrical Foundations
3. Fabrication Foundations
4. Software Foundations
5. Robotics Integration

Mechanical Foundations focuses on mechanical concepts like structural design, weight distribution, drivetrains, fastening, the relationship between speed and torque, and alternate methods of transferring motion such as linear slides and belts and pulleys. This track familiarizes students with the foundational skills needed to understand how components come together and different use cases for creating motion.

Electrical Foundations focuses on the foundational concepts around basic electricity and how circuits work. In this course, students learn how to use multimeters to measure various parts of a circuit to learn about concepts such as voltage and current. Students also learn how to control signals using a microcontroller (inputs and outputs), how to utilize binary sensors like Limit Switches, and analog sensors that provide a wide range of values like an Ultrasonic sensor. The culminating project is an e-panel consisting of all of the components found in a typical robotic system.

Fabrication Foundations introduces students to hand tools to cut, drill, and file down multiple materials to create a robot chassis and motor mount. It also introduces additive manufacturing (3D printing) to create a sensor mount. The skills taught include safety, basic measurement, reading and interpreting drawings, basic hand tool use, and handling materials.

Software Foundations is an introduction to programming concepts using one of the available popular robotics platforms (Arduino, LEGO EV3, VEX IQ, GoPiGo, etc.). Recipients of this certification demonstrate an understanding of the software engineering process through repeated planning, testing, and iteration throughout the units. Students also learn basic robot movement, how to use feedback from different kinds of sensors, and how to create complex programs using loops and decision-making logic.

Robotics Integration introduces learners to situations where technicians receive multiple components of a robotics system that require assembly, installation, and debugging. Students learn how to integrate components such as a vision sensor (camera) system, breadboard, servo motors, and embedded microprocessor from multiple hardware vendors. The learner will "unpack and test" components and refine “robot navigation programming" over the course of the units.

General across all courses

Attitudes tend to span across all courses. Below are the A’s that we have identified.

- Attention to detail
- Collaborating with others
- Creativity in problem-solving
- Re-use / efficient use of materials
- Safety
- Valuing workmanship
    - Robustness/reliability
    - Organization/neatness

Electrical Foundations

Students will be able to...	Applications
Construct simple circuits	● Construct wires with hand tools (S) ○ Cut and strip wire (S) ○ Crimp connectors onto wire (S) ● Identify conductors & insulators (K)

	• Select/attach connectors (S)
	    ○ Attach alligator clip onto wire (K)
	    ○ Use electrical tape to secure wires (K)

Troubleshoot and test wires	• Using Multimeters to test for Conductivity/Continuity (KS)
Control flow of electricity in circuits	• Integrate switches into an electrical circuit (KS)
Create and understand the differences between series and parallel circuits	• Construct a series circuits (K) • Troubleshoot a series circuit (S) • Construct parallel circuits (K)

Construct wiring panels using Terminal Blocks	• Understand how Voltage is distributed (K)
	    ○ Understand Voltage nodes (K)
	    ○ Understand and measure Voltage drops (KS)
	• Understand wire coloring conventions (K)
	• Using Multimeters to measure Voltage (S)
	• Select/attach connectors (S)
	    ○ Attach spade connectors (S)
	    ○ Use screw terminals (S)

Construct a circuit to power a motor using a coin cell battery	• Understand and use Moderate-voltage systems (KSA)
	    ○ Understand Safety with circuits (K)
	• Understand how Motors are powered (K)
	    ○ How battery size affects voltages (K)
	    ○ Understand torque and stalling of motors (K)

Measure voltage and current in a circuit	● Measure Current using a Multimeter (S) ● Understand current/amperage (K)
Select an appropriately sized battery	● Understand how a larger battery could affect voltage (K) ● Understand larger batteries and charge capacity (K) ● Safety warnings regarding higher voltages and current (K)
Adjust a circuit to alternate motor direction	● Flow/direction of current in a motor and how it affects the motor (K)
Add a Fuse to an existing circuit	● Understand the difference between AC and DC (K) ● Fuses and breakers ○ Purpose (K) ○ Identifying (KS) ○ Checking (S) ○ Installing/Changing (S)
Create a circuit that is powered by a microcontroller (or "brain")	● Programmable controllers (K) ○ Understand how to set up a basic micro-controller (Arduino Unos) (KS) ○ Understand and use General Purpose Input/Outputs (GPIO) pins (KS) ○ Plug into the system ground (KS) ● Use a programming software (KS) ● Select/attach connectors (S) ○ Header pins (KS) ○ Male/Female connectors (KS) ● Basic Programming (KS)

	○ Upload program to microcontroller (S) ○ Read comments in a program (S)
Change the rate signals are sent to a circuit	● Program basic signals (KS) ○ Arduino IDE ○ Blink ○ Comments ○ Parts of program ○ digitalWrite ○ pinMode ○ delay() + milliseconds ○ Uploading programs ○ Modifying programs
Incorporate Relays into the circuit to control motors	● Understand how Relays work (K) ● Understand how H-Bridges work (K)
Incorporate a Motor Control Board to control the direction of the motor	● Understand how to use a Motor Controller Board (KS) ● How multiple-voltage systems work (KS) ● Construct wires with hand tools (S) ○ Split paired wires ○ Three-wire connectors
Detect bumps using a limit switch	● Programming basic signals (KS) ○ Input signals ○ Output signals

	○ PULLUP Mode ● Understanding how sensors work (K) ○ Limit Switches
Detect objects from a distance using an ultrasonic sensor	● Understanding how sensors work (K) ○ Distance/Ultrasonic Sensors
Plan and construct the layout of an electronics board	● Electronics Organization/Planning (KS) ○ Planning Layout (S) ○ Labeling (S) ○ Wiring Harnesses & Conduits (S)

Mechanical Foundations

(KSA) Students will be able to...	Applications
Accurately measure component dimensions using a Measuring Tape / Caliper	● Measure linear distances (KS) ● Use measuring tapes (S) ● Use dial caliper (S) ● Imperial vs Metric units of length (K)
Assemble a basic structure following printed technical schematics	● Use of static structures (K) ● Read and interpret schematics (S) ● Identify chassis pieces and fasteners based on appearance and length (S)

	● Identify types of fasteners (K) ○ Bolts, screws, rivets, etc (K) ○ Differentiating bolts: threading, length, bolt head (button head, socket cap (KS) ○ Differentiating nuts: K-nut, lock nut, wing nut, etc (KS) ● Understand how to use hand tools (wrenches, screwdrivers, allen keys) to attach nuts and bolts (S)
Understand stability issues in the system and how to address them through bracing	● Problems with simple static structures (K) ○ Bending, breaking, collapsing (K) ■ Can move but not break ● Benefits of sturdiness and bracing in chassis/frame (K) ○ Adding a counterforce to support a structure ● Common types of bracing (K) ○ Diagonal, triangle, corner, lateral, etc (K) ■ What type of problem each type of bracing addresses (bowing, twisting, collapsing) ○ Materials used to brace (K) ○ Examples of bracing use ● How to use bracing (KS) ○ Bracing in 2 dimensions and 3 dimensions (KS) ● Identifying situations that can be improved by bracing (KS) ● Finding best place to mount (S) ○ Considering clearance and what else might be attached to the chassis in the future (KS)

	● Test bending/twisting of a structure before adding bracing (S)
Create an extruded chassis from the previously made square chassis	● Assemble an identical underside chassis using uprights, based off of provided schematics (S) ○ Differentiate pieces based off of length ● Fasten new chassis side to the pre-existing platform using simple uprights (S) ● Manipulate and push corners of chassis to find problems (S) ● Add bracing to solve bending/twisting/collapsing issues (S) ● Consider tradeoff between sturdiness and weight both in this situation and in the industry (KS)
Understand Balance and Center of Gravity	● Understand and find Center of Gravity (KS) ● Modify a structure to account for weight hanging off of one side of a structure (KS)
Understand the basics of how motors work	● Converting electrical power to mechanical power (KS) ○ How motors convert electrical to mechanical ○ Use of torque and rotational power in a motor
Construct a basic mechanical advantage device	● Construct a basic assembly using Exploded and Assembled diagrams (KS) ● Identify which component is generating the turning motion (KS) ● Identifying and understanding transmissions (KS)
Incorporate gears into an assembly	● Understand how a basic gear train transmits power and provides mechanical advantage (K) ○ Gear ratios (K)

	○ Use of gearboxes in drivetrains (K) ○ Mounting and using gear trains (S)
Incorporate chains and sprockets into an assembly	● Understand the difference between sprocket and gears (K) ● Construct a basic assembly using chains and sprockets (KS) ● Understand the mechanical advantage chains and sprockets provide (K)
Understand different drivetrains and their applications	● Construct different drivetrains and understand the reasons behind their applications (KS) ● Construct drivetrains and observe how wheel size affects movement (KS) ● Construct a 4-motor drivetrain and understand the benefits of 4-motor drive systems (KS) ● Incorporate omni-wheels into your drivetrain and understand the effects of that change (KS) ● Construct a mecanum drive system and understand the pros and cons (KS)

Fabrication Foundations

(KSA) Students will be able to...	Applications
Selecting an appropriate material for a robot chassis	● Understand the properties (Strength, Weight, Corrosion) of materials (K) ○ Plastics, Acrylic

Cutting materials based on those properties	● Basic measurement and interpreting drawings (KS) ○ Metric to US ● Read and interpret drawings (KS) ● Using precision measurement and layout tools (KS) ○ Calipers ○ Protractors and angle measurement tools ○ Combination Squares ○ Bench Vise
Drill holes in material in precise locations based on technical drawings	● Using hand tools (S) ○ Hand tool safety ○ Power tool safety ○ Experiencing the feeling of using hand tools on materials or fastening ○ Hand Drill ○ Hack Saw ○ Center Punch ● Basic measurement and interpreting drawings (KS) ○ Tolerances ● Add backing material to prevent cracking during cutting (KS)
Apply reductive manufacturing techniques (cutting and filing) to material based on technical drawings	● Using hand tools (S) ○ Coping Saw ○ Hand File assortment
Select material for a motor mount blank	● Identify proper aluminum material needed for project (KS) ● Utilize safe handling of material (S)

	• Measure proper sizing based on drawing (KS)
Construct a motor mount	• Use radius gauges to mark and create rounded corners (KS) • Filing down aluminum to size based on a diagram (KS) • Constructing a bending die (KS) • Bending aluminum based on diagram (KS) ○ Using a rubber mallet (S)
Set up and Maintain a 3D Printer	• Setting up the 3D printer on 3D printing software (KS) • Preparing start and end G-Code for a select printer (KS) • Understand nozzle and bed settings based on 3D printer (KS)
Open, view, and modify a 3D model in a modeling software	• Scale, and position model on 3D printing software to fit proper size (KS) • Specify print settings (KS) • View model on 3D printing software (KS)
Print a high-quality version of a 3D model	• Transferring 3D model to printer to print (KS) • Cleaning the model (KS)

Software Foundations

(KSA) Students will be able to...	Applications
Understand what software is (in a robotics context)	• Identify instances of, and purposes for, Software vs. Hardware (vs. Firmware) (K)

	- Understand the role of the programmer in the design of a robotic system (KA) - Be aware that, compared to mechanical or hard-wired functions, software can be changed quickly, easily, and cheaply -- and is therefore often used to improve the performance of systems even after they have been deployed through patches and other updates (KA)
Understand how software is used on a robot (Control logic)	- Software programs implement control logic for robots (KS) - Understand that the logic is separable from any particular software implementation (K) - Know that these logics may be expressed in words (pseudocode), diagrams (flow charts), and other forms in addition to code (K) - Read and reason about algorithms using pseudocode (S) - Read and reason about algorithms using flowcharts (S) - Typical control logic for a robot includes sensing the environment, planning a suitable response, then executing that plan (sometimes called SPA) (K)
Understand and write programs to implement simple control logic	- Understand the relationship between commands and parameters both syntactically (parts of a line of code) and semantically (meaning of each) (KS) - Read and program simple behaviors using (KS): - Open-loop command sequences (discrete or timing-terminated) - Closed-loop command sequences (sensor-terminated) - Conditional execution (sensor-contingent) - Loops and iteration

	■ Pure repetition (same commands n times) ■ Iterative repetition (for i=1..10 across something, not just counting) ■ Conditional repetition (while sensor<200) ● Create and prefer semantically useful functions (KSA)
Understand what input and output devices robot programs typically interface with	● **(Sensors)** Understand the physical principles behind common sensors, the kinds of input they would be expected to generate, their typical uses, and their typical failure modes (KS) ○ Internal state sensors ■ Rotational ("dial") potentiometers ■ Rotational encoders ■ Limit switches ■ Voltage sensors (battery meters) ■ Current sensors ■ Thermal probes ○ Touch-based sensors and buttons, including user interface devices ○ Passive optical sensors ○ Reflective rangefinding sensors: ■ Acoustic (sonar) ■ Optical (laser) ○ Vision: ■ Monocular (camera) ■ Stereoscopic (multiple cameras) ○ Inertial:

	■ Accelerometers ■ Gyroscopes ■ IMUs ○ Scanning vs. stationary sensors ● Know that robots that make effective use of sensors, especially multiple sensors working together (sensor fusion), perform better than those that do not (KA) ● **(Actuators)** Identify common output devices used by robots ○ Motors ○ Screens ○ LEDs ○ Speakers ○ Solenoids (including pneumatics and hydraulics)
Understand what a robot can and cannot reasonably do with programming	● Determine whether a suggested combination of sensing and output devices could feasibly accomplish a given hypothetical task (S) ● Be aware that certain types of tasks cannot be accomplished in real-time with a reasonable amount of computational power (K) ○ Combining information from more sensors takes more calculations, and also more memory to store the results ○ Inefficiently written code wastes computational cycles and memory
Understand software-related technician tasks	● Apply knowledge of sensors and programming logic to reason about potential causes of a performance malfunction in an installed system (KS) ○ Collect additional diagnostic data when appropriate (K)

	● Understand the purpose of common troubleshooting steps for robotic systems, e.g. power cycling, recalibration, updating firmware or software (KS) ● Follow instructions in a field guide for troubleshooting and repairing a deployed system (KS)

Robotics Integration

(KSA) Students will be able to...	Applications

Understand and articulate how robotics automation may be applied across broad contexts	● Identify and recognize different common industries in which robotics are applied (manufacturing, healthcare, agriculture, space, etc.) (K)
	● Recognize and give examples of robots doing the following major tasks: (KS)
	    ○ Locomotion (e.g. transporting boxes, self-driving vehicles)
	    ○ Manipulation (e.g. assembly line arm, microsurgery)
	    ○ Testing/Inspection (e.g. high-speed visual scanning)
	● Evaluate whether a robotic solution is appropriate or not for a given hypothetical scenario (KS)
	● Understand what specific kinds of tasks robots are used for
	    ○ Identify processes in an example that are more likely or less likely to be easily replaced by automation and robotics (KS)
	    ○ Identify the types of systems or subsystems would be needed to integrate robotics in a workflow (KS)

	● Know that some robots are designed from the beginning to be robots, while others are "add on" packages that modify the operation of other systems (K)
Understand the characterizations of a robotics system	● Know that categorizations of systems as robotic or non-robotic are imprecise, but generally, rely on certain characteristics (see below). Frame and compare technological systems with respect to these properties. (KS) ○ **Sensing:** Robotic systems collect information about the physical environments through sensors ○ **Perception:** Robotic systems process sensor data into useful intermediary forms that can be used in higher-level logic ■ Example: Identifying objects or their poses from vision data ■ Example: Composing a 3D voxel representation of the world from LIDAR data ○ **Planning:** Robotic systems decide a course of action based on an internal representation (model) of the world. Multiple layers of planning may be used to handle big-picture versus detailed decisions. ■ Example (high-level control): Calculating a path through a grid-world model of a room. ■ Example (low-level control): PID control over a motor to keep its speed constant. ○ **Acting:** Robotic systems produce a physical effect. This may result in the robot moving itself (locomotion), moving another object in the environment (manipulation), or directing something else (e.g. directing a laser beam to etch silicon in the manufacturing of microchips).

	● Compare and contrast representations of robots in popular fiction with real-world applications (KSA)
Understand the subsystems that comprise a robotics system	○ Explain why technical professionals usually divide complex systems into subsystems (K) ○ Know the differences between, and identify examples of, commonly used subsystems on robots, including Structural, Motion, Power, Sensing, Logic/Control, and Communications (KS) ○ Assemble the different subsystems of a robot and understand how the different subsystems come together to create the whole system (KS) ○ Understand how to assemble a robot's subsystems together to build a robot (KS) ○ Conduct triage on a hypothetical robotic system by identifying the subsystem in which a given problem likely originates (S)

Understand Control Logic that dictates the behavior of a robotics system	○ Understand the differences between full teleoperation (i.e. remote control), operator assistance or assisted teleoperation (e.g. remote micro-surgery), narrow autonomy in constrained environments (e.g. assembly line arm), and autonomy in more general environments (e.g. self-driving cars). (K)
	    ■ Identify cases of teleoperation, assisted teleoperation, and autonomy (KS)
	○ Understand the role of the programmer in the design of a robotic system (KA)
	○ What can a robot reasonably do with programming? What can it not do? (KS)
	    ■ Determine whether a suggested combination of sensing and output devices could feasibly accomplish a given hypothetical task (S)

Understand the purpose of different input and output devices and how they integrate into a robotics system.	● **(Sensors)** Understand the physical principles behind common sensors, the kinds of input they would be expected to generate and their typical uses. (KS) ○ Internal state sensors (e.g. Rotational encoders, Limit switches) ○ Touch-based sensors and buttons, including user interface devices ○ Optical sensors (Passive or Reflective) ○ Vision (e.g. Camera) ○ Inertial (e.g. Accelerometers, Gyroscopes, IMUs) ○ Scanning vs. stationary sensors ● **(Actuators)** Identify common output devices used by robots (e.g. Motors, Screens, LEDs, Speakers, etc.) ● **(Communications)** Know that robots may also receive (and send) information over wireless networks, and that this may enable additional modes of operation (K) ○ Teleoperation ○ Performing a primary function (data-gathering) ○ Multi-agent teams ("swarms")