Using ARC

Introduction

Robotics for everyone — no prior experience required

Building robots no longer requires deep programming or artificial intelligence expertise. Synthiam ARC removes those barriers by providing a platform designed for people at every skill level. Whether you are a hobbyist, an educator, or an experienced developer, ARC helps you create, test, and control robots without starting from complex code.

The platform combines a visual, block-based interface with a library of prebuilt components and device drivers. This lets beginners assemble reliable robot behaviors quickly while enabling advanced users to extend, customize, and integrate ARC with specialized hardware and workflows.

Visual programming and extensible functions

Instead of writing low-level syntax, you build behaviors by snapping together intuitive blocks and modules. ARC provides both general-purpose blocks for control flow and dedicated blocks for common sensors and actuators, so core robot logic is visible and easy to modify.

- Drag-and-drop blocks for control flow, sensor inputs, and actuator commands that make behavior design visual and predictable.
- Prebuilt integrations such as voice recognition and facial recognition to accelerate prototyping and add powerful features without extra setup.
- Extensibility to create custom functions or modules when your project needs specialized logic or to support unique hardware.

This approach shortens the learning curve and development cycle, making advanced robotics concepts accessible to non-programmers while preserving the flexibility that experienced developers require.

Reusable skills and the Skill Store

ARC organizes common robot capabilities into reusable modules called "skills." Skills encapsulate tested behaviors—such as navigation, object tracking, or user interaction—that you can install, configure, and combine to meet your robot’s objectives.

Synthiam’s Skill Store hosts contributions from community members and industry experts, enabling builders to share solutions, accelerate projects, and build on proven patterns instead of reinventing the same features.

- Browse a growing library of tested skills to reduce development time and risk.
- Mix, match, and customize skills to fit your robot’s hardware and goals.
- Join a community-driven ecosystem where contributors improve and refine shared solutions over time.

Overview video

Watch a short demo to see ARC’s visual programming and skills ecosystem in action.

Overview: Getting started with Synthiam ARC — a short demo of the platform's visual programming and skills ecosystem.

Interface & Workspace

Top Ribbon Menu

The ARC environment is controlled from the top ribbon menu, which remains visible at all times. The ribbon contains multiple tabs and menu items that group related functionality. Use the ribbon to access workspaces, add robot skills, configure settings, open manuals, and perform other global actions.

Top ribbon menu with tabs for project and workspace management.

Workspaces

An ARC project contains robot skills organized into workspaces. By default a project starts with one workspace; add more by pressing the **ADD** button. Workspaces let you separate different groups of skills (for example, navigation, vision, or dialog) so your project stays organized and easy to manage.

For detailed instructions on creating, renaming, and organizing virtual workspaces, see the manual:
Virtual Workspaces.

Robot Skills

Robot skills provide specific capabilities to your robot—examples include camera input, servo control, speech recognition, and response behaviors. A complete robot project is built from many skills placed into workspaces and configured to work together.

Learn more about available skills and how to use them in the Robot Skills documentation:
Robot Skills Overview.

Configure Robot Skills

Each robot skill includes a configuration menu for adjusting its behavior and parameters. Open a skill's configuration by clicking the configuration button (displayed as two dots ".." on the skill's title bar).

Click the configuration button on a skill's title bar to open its settings.

Robot Skills Help

Every robot skill includes a built-in manual with instructions and examples for setup and use. To open the manual for a skill, click the question mark (?) icon on the skill's title bar. The manual typically contains usage tips, parameter explanations, and links to additional resources.

Open the skill manual by clicking the question mark on the title bar.

Moving Robot Skills Between Workspaces

To move a robot skill to a different workspace, right-click the skill's title bar and choose the destination workspace from the context menu. The skill will be moved along with its current configuration intact. Use this feature to reorganize skills by category, project phase, or functional grouping.

Right-click a skill's title bar to move it to another workspace while preserving its configuration.

Auto-Arrange Robot Skills

By default, ARC automatically arranges robot skills within a workspace when a project is loaded or when new skills are added. You can disable auto-arrange for either case using the options presented in the corresponding dialog windows.

Loading a Robot Project

Dialog option to disable automatic arrangement when loading a project.
Uncheck this option to prevent auto-arrange when opening a project.

Adding a Robot Skill

Dialog option to disable automatic arrangement when adding a new skill.
Uncheck this option to keep manual control of skill placement when adding new skills.

Virtual Desktops

Virtual desktops (virtual workspaces) let you create separate desktop layouts, rename desktops, and move robot skills between them. This helps you organize complex projects and switch quickly between different task contexts.

For step-by-step instructions on renaming desktops, moving skills, and organizing desktops, see the Virtual Workspace manual.

Virtual Desktop Manual

Right-Click on the Desktop

Right-clicking anywhere on the desktop opens a context menu with shortcut commands that mirror many options from the top ribbon. From this menu you can add robot skills, switch desktops, view monitors, stop servos, and perform other common tasks quickly.

Right-click desktop menu with quick access to common commands and shortcuts.

How ARC Works

ARC running on a Windows PC and connected to an EZB I/O controller.

ARC runs on x86 Windows PCs, laptops, or single-board computers (SBCs)

ARC is designed to run on x86 Windows-based computers—desktops, laptops, or compatible single-board computers—so it can take advantage of a PC's greater processing power, memory, and available peripherals. ARC communicates with sensors, servos, and other hardware through a supported I/O controller called an EZB. Depending on the EZB model, the ARC PC connects to the EZB via USB or WiFi.

Running ARC on a PC lets you use native Windows drivers, standard USB devices (cameras, microphones, joysticks, storage), and more sophisticated algorithms that would be difficult to run on small microcontrollers. It also simplifies development and debugging using familiar PC tools.

Benefits and connectivity considerations

Using a PC for robot control provides several advantages, but the physical location of the PC (mounted on the robot or remote) changes how peripherals are attached and which connectivity options are practical.

Key benefits

- Higher compute capacity for complex programs, vision processing, and AI algorithms.
- Built-in support for standard USB devices and Windows drivers (cameras, audio, storage, input devices).
- Faster development and debugging with desktop tools and a familiar operating system.

Connectivity and deployment considerations

- If the PC is mounted on the robot, you can attach USB peripherals directly. This simplifies cabling and driver support.
- If the PC is remote, devices that require a local USB connection may need alternatives—networked cameras, USB-over-IP solutions, or attaching peripherals to the EZB or a companion computer on the robot.
- Consider USB cable lengths, power requirements, and available USB ports. Active USB hubs or powered extenders can help when distances or power are a problem.
- Driver compatibility on Windows is important—verify drivers for cameras, audio devices, and other peripherals before deployment.
- When using wireless connections, plan for latency and reliability: WiFi strength, interference, and network security can affect robot control and streaming data.

Example setup: ARC PC connected to an EZB controller with additional USB peripherals.

For a list of compatible EZB controllers and more hardware details, see the Hardware Overview: Compatible EZBs and hardware.

Project Menu

Overview

You can find the project menu within ARC in the top menu bar.
The menu items within the project menu are related to the project and saved with the project file.
These options enable personalization and custom adjustments of how the robot project will interact with hardware or the ARC software.
Items that you can select in this menu are 3D design, servo profiles, servo resolution, and add robot skills.

Robot Skills

Robot Skills can be added to the project by this menu option. Custom skills can be created as well. Read more about what a Robot Skill is by clicking here.

Add

Press this menu option to display the Robot Skill menu and select the robot skill to add to the current project workspace.

Create

This menu option allows creating a Robot Skill project template. The menu will be displayed where you can enter information about the robot skill that you wish to create.
A Visual Studio project will be generated with the information you entered, including a registration on the Synthiam cloud server under your user credentials.

Auto Arrange Robot Skills

By default, robot skills are auto arranged on workspaces when either a project is loaded, or a new robot skill is added to a workspace. This behavior can be disabled for either scenario by unchecking the option in the respective dialog window.

Loading Robot Project
Adding Robot Skill

My Robot

The My Robot menu category is a specific option for the robot project. These include 3D design files, 3D instructions, and project properties.

View
Add a robot skill that displays the current 3D designed robot. Robots can be designed to include the CAD files with a project using the Design button in this menu category.

Instructions
Open a full-screen window that displays the automated instructions to assemble the robot using the 3D CAD files.

Design
Use this menu option to design or modify a robot with 3D CAD files. This is how you build a robot to be displayed in the instructions to be reproduced.

Properties
Allows specifying the project-specific properties, such as servo resolution, project size, thumbnail, default robot skill, and other options.

Press the Project Size button on the properties page to view the project size. This will display the configuration size of each robot skill. It will help you narrow down unused configurations that are unnecessary to lower the project file size.
Audio is usually the largest configuration size. It is recommended to CROP or TRIM the audio for only the portion that you require to play. For example, if a 60 second MP3 is added to the project but only the first 4 seconds are played, trim the remaining 56 seconds.

3D Parts

Before 3D CAD files can be shared with other users, they must be uploaded to the Synthiam cloud server.
The options in this section allow you to upload custom CAD files to the cloud or view and synchronize
your existing library.

Library

Opens a new window with tools for managing your local CAD design library.
From this window, you can synchronize your local CAD cache with the Synthiam cloud server
and upload custom CAD files that you have created using the Bit Builder.

Bit Builder

Share your 3D design files with the Synthiam ARC community by publishing them as EZBits.
The Bit Builder can be accessed from the **Project → 3D Parts → Bit Builder** menu.
This tool allows you to create custom 3D CAD files for use in the 3D Robot Designer.
Once your design is complete, it can be uploaded to the cloud using the Library menu option.

When you first open the Bit Builder, you will be presented with an instructional introduction page
that explains the basic workflow.

The **Design** tab allows you to begin designing your EZBit.
An EZBit can consist of one or more STL design files.
You can add STL files to the 3D designer and position them as needed to create the final assembly.
If a part is included only for visual reference (for example, a servo shown as part of a servo mount),
you can mark that part as *not 3D printable*.
Non-printable parts are used purely for visualization when others design their robots using your EZBit.

After completing the design, you can define detailed properties for the EZBit so other users understand
what it is, how it is used, and how it should be printed.
The **Properties** tab includes the following options:

- EZBit name
- Description of the EZBit and its purpose
- Capture a snapshot from the Design tab to use as a thumbnail
- Category assignment
- 3D printer type
- Suggested print settings, including layer height, shell count, infill, material, and estimated print time based on your configuration

The Properties tab also includes an **EZ-B Connections** section.
This allows you to specify how the EZBit connects to an EZ-B controller.
For example, if the EZBit is a servo-based part, you can define a servo connection.
This information helps users track port usage when assembling robots in the My Robot Designer.

To upload your EZBit, first save it to your local library.
Then open the EZBit Library Manager and navigate to the **Upload** tab.
Press **Refresh Library** to display all EZBits you have created.
Select the EZBit you wish to share and click **Upload Now**.
Once uploaded, your EZBit will be available for the entire Synthiam community to use.

Servo Profile

When a robot project is used across multiple physical robots, the servos may not be aligned exactly the same. In these situations, servo offsets can be fine-tuned to compensate for small differences in alignment. For example, one robot may have a servo centered perfectly at 90 degrees, while another identical robot may center at 87 degrees. In that case, an offset of -3 would be applied.

Important: The values in this menu are **offsets**, not servo positions. Offsets are typically only a few degrees for small calibration adjustments.

If you find that a servo requires a large offset, such as 5 to 10 degrees or more, it is usually better to physically adjust the servo joint instead. This may involve removing the servo horn screw, repositioning the bracket, and reattaching it so the joint is properly aligned when the servo is set to its intended neutral position, such as 90 degrees.

There are two ways to configure a servo profile. The **Configure** option is used when an **Auto Position** robot skill defines the servo layout. If no Auto Position skill is available, the **Advanced** option can be used to adjust all servos across all EZ-B controllers manually.

Configure Button

The **Configure** option uses the servo layout from the **Auto Position** robot skill. This provides a more user-friendly calibration experience because the servos are shown in a visual layout that represents the robot. This makes it easier to identify which servo is being adjusted.

If a 3D robot model is not included in the project, you can also add a robot image to help identify the servo locations visually.

Preparing Your Custom Robot For Fine Tuning

When creating a servo profile for a custom robot, it is recommended to first define a default Auto Position frame, or create a calibration frame, that places all servos in a repeatable position such as 90 degrees. Because a servo profile stores offsets rather than positions, the robot must always begin from a known and reproducible pose. This gives the servo profile a consistent reference point for fine-tuning.

The purpose of the servo profile is to allow users to make small alignment corrections so the robot reaches the intended pose even if servo installation or servo model tolerances differ slightly from one robot to another.

Since the on-screen servo placement is pulled from the Auto Positioner, it is recommended to place each servo marker close to the actual joint it affects. Proper spacing and placement make it much easier to understand which joint each offset is modifying.

Advanced Button

The **Advanced** option is useful when the project does not include an **Auto Position** skill, a 3D model, or a robot image. Instead of displaying a visual layout, this option shows a complete list of all servos across all EZ-B controllers so they can be adjusted directly.

This mode is ideal for custom projects or technical setups where a visual representation is not available.

Save and Load Servo Profiles

Servo profiles created with either the **Configure** or **Advanced** option can be saved and loaded later. Profiles may be stored locally on the computer or saved to the cloud, making it easy to reuse calibration settings across projects or devices.

Remote UI

The ARC Remote UI option allows mobile devices to operate the robot remotely using custom interfaces you have created.
You can add as many interfaces as you'd like using the Interface Builder robot skill.
The interfaces can have multiple pages that are switched by a button.
These interfaces are displayed on an Android or iOS mobile device using the ARC Remote UI app.

Interface Builder Robot Skill

This feature uses interfaces built with the Interface Builder robot skill. The Interface Builder empowers you to create your robot's touch-screen user interface (UI). Use buttons, labels, pads, sliders, drop-downs, checkboxes, and displays to make a control panel to activate features.

Detailed information on the Interface Builder manual page will help you configure a custom interface with Remote UI.

Where To Find The Remote UI Option

Configure the option to enable the Remote UI Server by accessing the menu from the top menu in ARC.
Select the *Project* -> *Properties* option to display the properties menu for your robot.
This option is configured per robot.

Configuring Remote UI Server

This tab of the robot project properties will display the options for enabling and configuring the Remote UI server.
When configured, this will apply to the current project and is not ARC system-wide.

This means that configurations in this menu are saved with the robot project.

Enable Remote UI Server

This checkbox determines whether the Remote UI server will be active for the current robot project. The Remote UI Server will start when the project loads and this checkbox is checked.

TCP Port

The TCP port is 3184 by default. However, you can change that for your network preferences.
Optionally if you desire connectivity to your robot from the internet, this is the port you would forward through your router configuration.

Remote UI Password

This is a mandatory field to complete.
Setting this option prevents access to your robot without this password.
You will be prompted for this password when connecting to the robot from the ARC Remote UI app on your mobile device.

Compression Quality

This option will configure the compression level when sending graphics, video, and other media elements.
A higher number will be higher quality and much larger transmission sizes.
A lower number will be lower quality (grainy) images but faster and smaller transmission sizes.
Configure this value accordingly for the best response and image quality if connecting over the internet, specifically using a mobile data plan.

Update Frequency

This setting determines how often to send updates to the mobile device in milliseconds.
Configuring this value too low will result in high bandwidth and CPU usage on the ARC computer.

Broadcast

When the Remote UI server is enabled, ARC will broadcast itself on the local network.
Any mobile device with the ARC Remote UI app will see the ARC instances being published, including the IP Address, Port, and computer name.

Mobile App

The mobile app is currently available for Android mobile devices at this time. You will find the link below to get it from the Google Play Store.

***Note:** *The iOS app is currently under review by Apple and will be available once Apple has allowed it. (updated Nov 18, 2022)*

Creating Interfaces

The Interface Builder robot skill is used to create interfaces that will be displayed to the Remote UI clients.
Multiple interfaces can be defined, and switching interfaces can be done using the *ShowControl()* command in all ARC programming languages.
The Cheat Sheet tab will display the ShowControl() options.
Only Interface Builder robot skills can use the ShowControl() option.

The first interface loaded for Remote UI clients will have checked the "*Make this the default Remote UI & fullscreen interface*" option.

View Interface Builder Manual

Viewing Remote UI Log

The Remote UI logs connections and disconnects in the ARC Debug Log tab. This tab is accessed from the top File Menu in the virtual desktop section.

Connecting Over Internet

Using the local IP address of our computer with a mobile device is easy because they are on the same network.
However, a few steps are needed if you wish to connect your mobile device over the internet to a robot in your home behind a router.
Specifically, you will need to have your router/firewall forward the port from the internet to the ARC PC.
Every router is different, so we cannot provide step-by-step instructions.
The result and terminology will be the same. The feature will be called Port Forward.
The port that Remote UI uses by default is 3184.
The broadcast feature, where the robots show up on your mobile device, will not be possible when using the app over the internet.

How to use ARC with Microcontrollers

Table of Contents

1. Introduction
2. Limitations of ARC on a PC for Closed-Loop Functions
3. Utilizing Microcontrollers for Closed-Loop Functions
4. Examples of Closed-Loop Functions

Inverted Pendulums with Multi-Axis Accelerometers
Counting Encoders for Positioning
Monitoring Current Consumption for Servo Grippers

5. Microcontroller Selection and Configuration
6. Writing Code for Closed-Loop Functions
7. Integrating Microcontrollers with ARC Robot Software
8. EZB Protocol
9. Conclusion

Introduction

This guide explains how to use microcontrollers to implement tight closed-loop control while working with the ARC Robot Software (Synthiam). ARC runs on a PC and is excellent for high-level behavior, UI, and visualization, but general-purpose operating systems cannot guarantee the low latency and deterministic timing required for some closed-loop tasks. Offloading those real-time control functions to microcontrollers improves responsiveness, accuracy, and system modularity. This document highlights the benefits, common use cases, hardware and software considerations, and how to integrate microcontrollers with ARC.

Limitations of ARC on a PC for Closed-Loop Functions

PCs and desktop operating systems are not real-time environments: task scheduling, background processes, and OS jitter can introduce unpredictable delays. For closed-loop control systems that require frequent sampling, precise timing, and low latency (for example fast PID loops or encoder counting for motion control), this unpredictability can degrade performance or cause instability.

Typical problems when running tight loops on a PC include:

- Variable loop timing (jitter) caused by OS scheduling.
- Latency introduced by USB, serial, or network communications.
- Limited access to hardware timers, interrupts, and high-resolution peripherals.

Using microcontrollers for these tasks preserves ARC for higher-level coordination while guaranteeing deterministic local control.

Utilizing Microcontrollers for Closed-Loop Functions

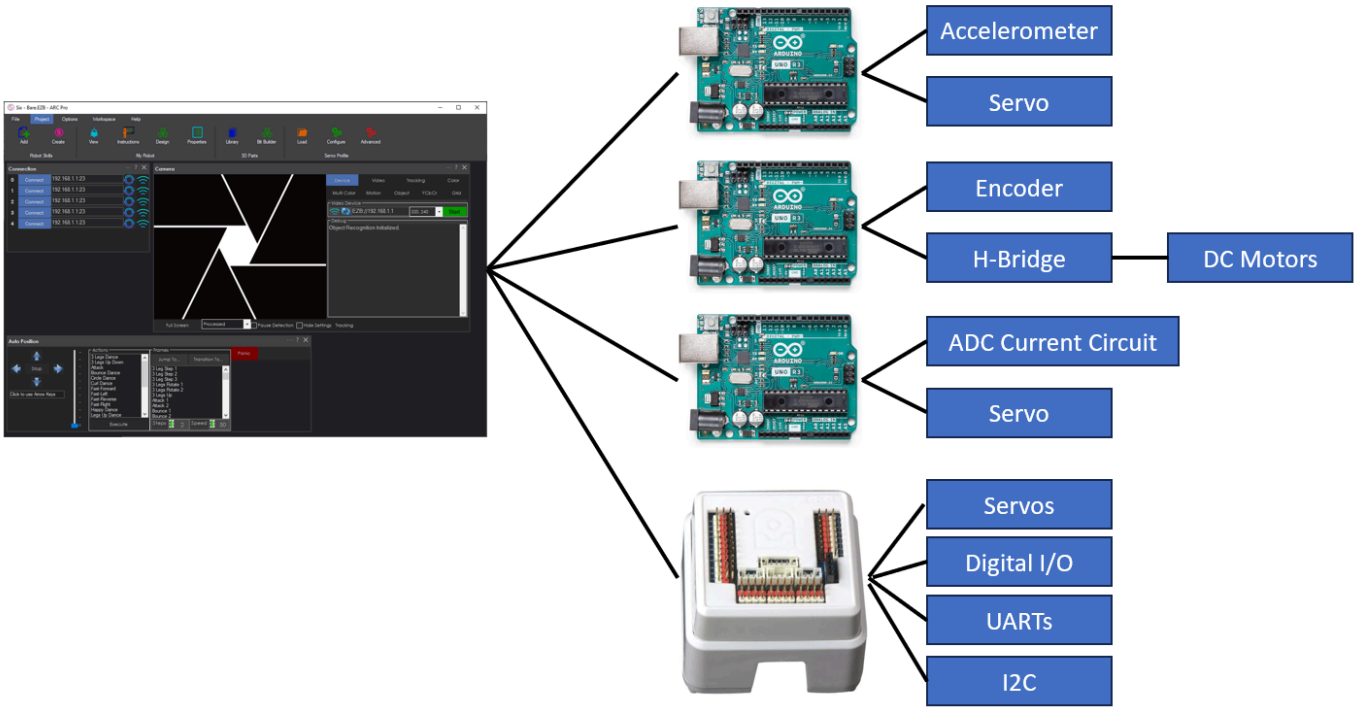
Microcontrollers are purpose-built for real-time tasks. They provide low-latency I/O, hardware timers, interrupt handling, ADCs, and peripherals such as quadrature encoder inputs. Distributing control across multiple microcontrollers can be both cost-effective and scalable.

Key advantages

- **Deterministic real-time processing:** hardware interrupts and timers allow reliable sampling and control intervals.
- **Higher effective resolution and accuracy:** dedicated controllers can poll sensors and run control loops at high rates.

• **Cost efficiency and modularity:** many inexpensive microcontrollers can be deployed to handle specific functions, reducing load on the PC.

A common pattern is to flash each microcontroller with a small custom firmware (often referred to as a custom EZB firmware) that performs the closed-loop task and communicates only essential data to ARC. Where appropriate, the EZB communication protocol can be extended so firmware both sends telemetry to ARC and accepts high-level commands. If desired, multiple functions can be consolidated into a single microcontroller, but often a one-function-per-controller design simplifies development and debugging.



Distributed microcontrollers (each with custom EZB firmware) perform closed-loop control and optionally exchange data with ARC.

Examples of Closed-Loop Functions

The following examples illustrate typical closed-loop tasks that benefit from microcontroller-based implementation.

Inverted Pendulums with Multi-Axis Accelerometers

An inverted pendulum requires fast, continuous feedback to remain balanced. A microcontroller can read multi-axis accelerometers (and gyroscopes), run a PID or state-space controller at a high sampling rate, and drive actuators with low latency. Offloading this loop to a microcontroller reduces jitter and improves stability.

A practical example and corresponding firmware are available in the Inverted Pendulum robot skill: Inverted Pendulum robot skill. That skill demonstrates offloading the closed-loop PID to a microcontroller and exchanging status and commands with ARC.

Counting Encoders for Positioning

Quadrature encoders and fast counters are best handled by microcontrollers or

dedicated encoder hardware. Microcontrollers can perform interrupt-driven counting, apply debouncing or filtering, and calculate velocity or position at high rates, then report counts or filtered values to ARC for navigation and higher-level decisions.

Monitoring Current Consumption for Servo Grippers

Real-time current monitoring lets a controller detect stalls, overloads, or changes in grip strength. A microcontroller can sample current sensors, apply thresholds or adaptive logic, and immediately cut or adjust drive signals for safety. ARC can then log events and make higher-level decisions based on reported telemetry.

Microcontroller Selection and Configuration

Choose a microcontroller that matches your functional needs and interface requirements. Common choices include Arduino (AVR, SAMD), ESP32, STM32, and specialized encoder/counting or motor-control MCUs.

Selection considerations

- **Processing power:** required loop frequency and algorithm complexity (e.g., state-space vs PID).
- **Peripherals:** ADC resolution, hardware timers, interrupt capability, quadrature encoder inputs, PWM channels.
- **Communication:** UART/Serial, I2C, SPI, CAN, or USB for integration with ARC.
- **Memory and libraries:** available flash and RAM for firmware and buffering.
- **Power and form factor:** voltage domains and board size constraints.

Configuration involves setting up the programming toolchain, pin mapping, peripherals (timers, ADCs), and robust communication parameters (baud rate, packet framing, retries). Where precise timing is required, make use of hardware timers and interrupts rather than polling loops.

Writing Code for Closed-Loop Functions

Firmware for closed-loop control typically includes sensor acquisition, filtering, the control algorithm, actuator output, and communication. Follow best practices to ensure reliable real-time behavior.

Recommended practices

- Use interrupt handlers and hardware timers for deterministic sampling and encoder counting.
- Keep control loops lightweight and deterministic; use fixed-point arithmetic if floating-point is too slow.
- Apply appropriate filtering (low-pass, complementary, or Kalman where needed) to sensor signals.
- Provide telemetry and diagnostic messages to ARC but avoid sending high-frequency raw data unless necessary.
- Implement safety features: timeouts, watchdogs, overcurrent protection, and graceful failover.

Leverage existing libraries and examples for sensors, encoders, and motor drivers. Test control loops thoroughly with simulation or slow speeds before running at full performance.

Integrating Microcontrollers with ARC Robot Software

Integration typically means establishing a robust, low-latency communication channel between ARC and the microcontroller and defining a compact protocol for commands and telemetry.

Common integration options

- **Serial/UART:** simple, reliable, and widely supported; use framing and checksums for robustness.
- **I2C / SPI:** for short, local buses between boards; often used for sensor networks.
- **CAN:** for multi-node networks with robust bus arbitration in noisy environments.
- **Wi-Fi / Ethernet:** for remote devices (e.g., ESP32); consider latency and packet loss.

Design your data exchange so ARC receives only the necessary state updates (e.g., position, error codes, or aggregate telemetry) while the microcontroller retains responsibility for high-rate sampling and loop stability. Example: the Wheel Encoder Counter robot skill demonstrates a custom EZB firmware that extends the protocol so ARC can query encoder counts from the microcontroller: Wheel Encoder Robot Skill.

Video: Demonstration of the Wheel Encoder Counter skill and extended EZB firmware communicating encoder counts to ARC.

EZB Protocol

The EZB communication protocol defines commands and message structures for interfacing with peripherals (UARTs, I2C, digital I/O, PWM servos, etc.). The protocol documentation is available here: EZB communication protocol. The protocol can be extended to add custom commands and telemetry as demonstrated by the Wheel Encoder example.

When extending the protocol, keep message formats compact, versioned, and backwards compatible. Implement basic error detection (checksums) and sensible retry/backoff logic in both ARC and the firmware.

Conclusion

Offloading tight closed-loop control to microcontrollers improves determinism, accuracy, and safety while letting ARC remain the high-level orchestrator. This approach enables scalable, cost-effective systems that combine the strengths of both PC software and embedded controllers. To start creating a robot skill that integrates with microcontroller firmware, see the Robot Skill Tutorial: Robot Skill Tutorial.

Robot Skills

What is a Robot Skill?

ARC apps consist of Robot Skill Controls. Each skill is a behavior for the robot, similar to a process (or node), and they run simultaneously in their own thread at the same time. Robot skills can communicate with each other using the ControlCommand() command. There are skills for Cameras, Speech Recognition, Artificial Intelligence, and hundreds more. Robot Skills are added to the ARC project workspace using the Add Skill option located in the Project tab of the main menu.

By combining multiple skills, robots can perform advanced and complex tasks. Robot skills come in two flavors...

1. **Built-In Robot Skills -** These skills developed by Synthiam are included with ARC installation and cannot be removed.
2. **Plugin Robot Skills -** Developed by third parties and may be added/removed from the ARC software through the Synthiam Robot Skill Store.

Here is a screenshot of an example ARC project demonstrating multiple robot skill controls. Each skill control performs a specific function of the robot. An ARC Pro user's projects may contain an unlimited number of skills (as PC memory allows). Read more about robot skills here.

Add Robot Skill to Project

Robot Skill Controls can be added to your project using the Add Skill option located in the Project tab of the main menu.

Add Robot Skill Menu

When the Add Skill button is pressed, the dialog will display a list of all skills in their respective category. Selecting a skill will add it to your project. Some skills will be grey because they have not been installed. Press the download icon on the skill when an internet connection is available to download and install the robot skill.
The *Auto Arrange* checkbox at the bottom of the Add Skill dialog will auto-arrange all robot skills on your project to fit the new skill if checked.

Installing Robot Skills

Synthiam ARC includes several robot skills by default.
However, as we grow new partners, new robot skills will be available to you.
When a robot skill is available but not yet downloaded, it will have a Download icon next to it.
With an internet connection and clicking the icon, you will be prompted to download the robot skill and install it automatically.
Or, you may view the manual to learn how it works.

Uninstall Robot Skill

Uninstalling a robot skill removes the installation files from the hard drive.
In the Add Robot Skill dialog, locate the robot skill you wish to uninstall and right-click on the icon.

Uninstall After Clean Startup

The robot skill must not be in use when uninstalling. Otherwise, an error will be displayed.
If you have recently used the robot skill, you may be prompted to close and reload ARC to uninstall the robot skill.
Reloading ARC to delete robot skills will be necessary when the operating system caches the library and locks the file.
Closing ARC will remove any references to the library. The next time ARC is loaded, the robot skill can be uninstalled as long as you do not load the robot skill again.

Screenshot

In the Add Robot Skill menu, right-click on the robot skill that you wish to uninstall. Select Uninstall from the drop-down.

Auto Arrange Robot Skills

The auto-arrange feature in ARC will re-organize robot skills to fit the workspace when robot skills are added to a project, or an existing project is opened.
By default, this setting is enabled to auto arrange robot skills on workspaces. This behavior can be disabled for either scenario by unchecking the option in the respective dialog window.

Loading Robot Project

When loading an ARC robot project, the dialog window has a checkbox in the lower left to configure auto-arranging robot skills.

Adding Robot Skill

When adding a new robot skill to an ARC project, the dialog window has a checkbox in the lower left to configure auto-arranging robot skills.

Organize Robot Skills (virtual dekstops)

As a robot project grows, it will have several robot skills that clutter the workspace.
Synthiam ARC can customize virtual workspaces to organize robot skills.
The workspaces are located in ARC's top menu under the File tab. Pressing the ADD or

REMOVE buttons allows you to add or remove workspaces—Right-click on a workspace to rename it.

In this screenshot example, we have organized the robot skills based on their function with the robot.
The Input workspace hosts interactive robot skills, such as joysticks and speech recognition.
The Camera workspace hosts the robot's camera and various tracking and computer vision robot skills.
The Processing workspace hosts scripts and AI Chatbot robot skills.
Finally, the Movement workspace hosts the robot skills responsible for moving the robot and interacting with the real world.

See the Virtual Workspaces manual for more information.

Virtual Workspaces Manual

Syncing and Updating Robot Skills

Learn how ARC synchronizes robot skills and updates.

Note: An active internet connection is required for ARC to synchronize with Synthiam servers.

What is a Robot Skill?

A Robot Skill is a pre-programmed functionality that you can add to your robot through Synthiam ARC.
Skills allow your robot to perform specific tasks such as controlling servos, reading sensors,
recognizing objects, playing sounds, or interacting with users.

Robot Skills are modular, meaning you can mix and match them to build complex robot behaviors without
writing code. For a full overview, visit the
Robot Skills Overview.

How Does ARC Sync Robot Skills?

Each time ARC starts, it checks whether your computer has an active internet connection. If a connection
is available, ARC securely communicates with Synthiam’s servers to:

- Retrieve a complete list of all available robot skills.
- Check for updates to any robot skills already installed.
- Download and apply updates so you always have the latest features and fixes.

This automatic synchronization ensures your robot skills stay current without requiring manual downloads.

Steps to Ensure ARC Syncs Properly

1. Confirm your computer is connected to the internet.
2. Launch ARC.
3. Allow a few seconds for the synchronization process to complete.
4. Open the Robot Skills Manager in ARC to verify the available skills and updates.

Forcing the Robot Skill List to Update

In rare cases, you may want to manually force ARC to refresh its list of available robot skills.

This can be done by removing the cached skill list file.

1. Ensure ARC **is not running**. Close ARC completely if it is open.
2. Verify you have an active internet connection by opening a web page (for example, `synthiam.com`) in your web browser.
3. Open File Explorer and navigate to `C:\ProgramData\ARC`.
4. Delete the file named `AvailablePluginsList.txt`.
5. Restart ARC.

What this does: The `AvailablePluginsList.txt` file stores a cached list of robot skills downloaded from Synthiam's servers. When this file is missing and ARC starts with an internet connection, ARC automatically downloads a fresh, up-to-date list.

Can't Find the ProgramData Folder?

By default, Windows hides the `ProgramData` folder. If you do not see it, you will need to enable the option to show hidden files and folders.

1. Open **File Explorer**.
2. Click the **View** tab at the top.
3. Check the box labeled **Hidden items**.
4. Once enabled, the `ProgramData` folder will become visible, and you can navigate to `C:\ProgramData\ARC`.

Tip: You can also paste `C:\ProgramData\ARC` directly into the File Explorer address bar and press Enter.

Help & Manual for Robot Skills

Every skill control has a question mark next to the close X button. Pressing the question mark button will direct you to a manual page for the respective skills.

Version & Manual

With plugin robot skills, you can get the version and manual by clicking the ? (question mark) on the robot skill in the project. If this is a built-in robot skill, you will be brought to the manual because there is no version. Only plug-in robot skills contain versions.

Configuring Robot Skill Settings

Configure Button

Every robot skill control includes a configure button in its title bar. Clicking this button opens the configuration dialog for that specific skill, where you can adjust the options that control how it behaves.

The configure button opens the skill's configuration dialog.

Robot Skill Name

Each skill window has its own configuration dialog with settings related to that skill's behavior. One setting is shared across all robot skills: the Skill Title or Name. Every skill control must have a unique name.

Since **ControlCommand()** references robot skills by name, you can rename a robot skill when needed. To do this, right-click the robot skill title bar, select **Rename**, and then enter a new name.

A unique name is important because the **ControlCommand()** function can send commands to skills and change parameters programmatically. For example, you can use JavaScript to tell the Camera Skill to enable face tracking. The **ControlCommand()** function is available in Python, JavaScript, and EZ-Script. In Blockly, you can find **ControlCommand** under the **Utilities** tab.

Each robot skill includes a configuration dialog with a unique skill name.

Robot Skills Messaging (ControlCommand)

Skill Messaging (ControlCommand) Example

Robot skills are multithreaded, which means they run at the same time in separate threads. Using ControlCommand(), one robot skill can send a command to another robot skill. For example, a speech recognition skill can instruct a camera device to begin face tracking. Skills can also send commands to each other programmatically. In this example, speech is detected and events are passed between skills to produce the desired robot behavior. This type of interaction can also be recursive if needed. To send commands, use the ControlCommand() script method. This command is available in the programming script interfaces of supported robot skills, allowing one skill that supports scripting to instruct another skill to perform an action.

Renaming a Robot Skill

Because `ControlCommand()` references robot skills by name, you can rename a robot skill when necessary. To do this, right-click the robot skill title bar, select **Rename**, and enter a new name.
Read more about using `ControlCommand()` in script programming:
Control Command Details

Example of ControlCommand skill messaging between robot skills.

Create Custom Robot Skill

Custom robot skills can be created and published in Synthiam's Skill Store for public or private use within your organization. There is a great step-by-step tutorial on making a control HERE.

Options Menu

Overview

The options menu of ARC provides methods of changing how ARC operates, colors, and account management. Press the *OPTIONS* menu tab to access the list of menus.

Debug

The DEBUG menu option will permanently open the debug popup at the bottom of the ARC workspace. The debug popup normally is displayed when new messages are added to the debug log. By pressing this button, the debug window will always be displayed.

To have the debug menu stop being permanently displayed, uncheck the Always Show option.

Twitter

Some ARC robot skills have the capabilities to interact with a Twitter account. You can create a custom account for your robot, or use your personal account. The robot skills can either read tweets, so you can remote control it via Twitter - or post to Twitter, such as new images or activity.

Twitter Account Settings

Step 1 - Obtain Twitter Verification code. Press this button to open the Twitter website, log in, and grant permission for Synthiam ARC to have access to the Twitter account. You will be given a verification code. Use this code for the next step.

Step 2 - Enter the verification code. This code was provided to you from the previous step

Completed - You may now use robot skills that post or read from Twitter with your robot.

Nest

Some ARC robot skills can control a Nest thermostat. The Nest thermostat will provide you with a verification code that can be entered here and saved for the current Windows user.

Step 1 - Obtain Nest Pincode. Press this to be directed to the Nest website. Login with your

account and agree to give Synthiam ARC access. You will be given a pin code, which will be used in the next step.

Step 2 - Enter Pincode. Enter the code from the previous step

Preferences

The preferences menu can be accessed to change how ARC looks and feels. This menu has several options; each is documented and explained in further detail by hovering over the question mark next to the respective option.

General Preferences

- **Full Screen**:
Enable this option to run the ARC application in full-screen mode.
- **Debug Show Time**:
Set the duration for which the debug messages are displayed in seconds.
- **Do not show Debug Popup**:
Check this option to turn off the debug popups from appearing.
- **Adjust Drag Sensitivity**:
Set the sensitivity level for dragging actions within the application. You can also specify a different sensitivity when the shift key is held.
- **Prompt for Servo Profile when loading projects**:
Enable this option to be prompted to select a Servo profile when loading projects.
- **Override and Always prompt for Servo Profiles when loading projects**:
Check this option always to be prompted to select a Servo profile, overriding any default settings.
- **Do not display exit confirmation when exiting**:
Enable this option to skip the confirmation dialog when exiting the application.
- **Show Tutorial Link Window when ARC loads**:
Check this option to display the tutorial link window each time ARC loads.
- **Default EZ-Script edit mode**:
Select the default script editing mode from the dropdown menu. Options include Javascript.
- **WaitForSpeech() minimum confidence value**:
Set the minimum confidence value for the WaitForSpeech() function. This value is a decimal between 0 and 1.
- **Do not prompt to remove controls from project**:
Enable this option to skip prompts when removing controls from a project.
- **Enable Auto Backup**:
Enable this option to automatically backup your project. You can set the interval in minutes.
- **Auto arrange skill padding**:
Set the padding value for auto-arranging skills within the application.
- **Enable auto subscription refresh**:
Check this option to automatically refresh your subscription status.

Proxy

The proxy menu allows configuring how ARC will interact with a proxy server on the network. If you prefer, you can configure the Windows System Proxy settings, overriding this setting. It is easiest for educational institutions or large networks not to use this menu

and configure the proxy across Windows using the Windows option.

Bit Designer

These are the colors for the bits and the interface when designing robots using 3D printable bits or viewing the automated instructions to build a robot. These colors are used twice: when building a robot in the Robot Designer or viewing the Robot Build Instructions.

Window Theme

ARC colors can be adjusted for a specific theme look and feel. The default color scheme is DARK, designed to be energy efficient and provide eye strain relief when using ARC for long periods. When ARC is first loaded, you are prompted to select a default scheme. The scheme you have chosen can be altered here.

Audio

This is an advanced menu for configuring EZB audio settings. These values should not be changed, or EZB audio streaming will be affected. If you are unsure of the values, press the Defaults button or change the pre-buffer size to 20000 and packet size to 256.

Advanced

The advanced tab has several options for performance features and various settings. It is recommended that you read the question mark popups for the possibilities. This page is for advanced users.

Remote Access Server

The remote access server enables sharing the PC running ARC, including streaming audio in both ways. This allows using a remote device (e.g., Android or Chromebook) to use the ARC computer and audio resources. The client's mic will be routed to the ARC PC, and the ARC PC audio will be routed to the client. Read the manual HERE for more details about the Remote Access Server.

Remote Access Service

The Remote Access Service enables mobile clients to access the desktop of the PC running ARC. This unique client/server system routes audio between the client and server, allowing you to use the microphone on your mobile device as a remote mic for the ARC PC and the speaker on the remote device as a remote speaker for the ARC PC. Additionally, it provides screen-sharing functionality similar to that of a remote desktop.

Index

- Why Use Remote Access Service?
- Terminology
- Downloads
- Network Configurations
- Using the Remote Access Client
  - Server Selection Screen
  - Remote Control Screen
  - Audio Redirection
- Audio Redirection Setup Instructions
- Enabling Remote Access Server in ARC
- Screen Scaling
  - Automatic Selection of Screen Capture Method
  - How to Check Your Screen Capture Mode
  - Setting Screen Scaling in Windows
- Diagnosing Remote Access Server
- Security Recommendation

Why Use Remote Access Service?

• If your robot has an SBC onboard, allow for remote control without 3rd party applications such as VNC or Remote Desktop.
• In educational institutions, devices like Chromebooks, tablets, or iPads can now provide the full ARC experience.

Terminology

• **Remote Access Service (RAS)**: Used to describe the client/server as a whole.
• **Remote Access Service Server (RASS)**: Describes the server feature within ARC that accepts client connections for desktop and media sharing.
• **Remote Access Service Client (RASC)**: An application that runs on a client device, such as a Chromebook or Android device, that establishes a connection to a RASS (Remote Access Service Server).

Downloads

The RASC (Remote Access Service Client) is currently only available for Android devices through Google Play. We recommend using this app on a device with a large screen, mouse, and keyboard, such as a Chromebook.

Network Configurations

Your robot will require a dedicated PC, which can be as cost-effective as an SBC. The SBC will need one of the following network configurations:

1. **Single WiFi & Ethernet:** The robot operates in Adhoc mode, with the SBC connecting to the robot's WiFi and the Internet via Ethernet. The Remote Access client can connect to the WiFi or Ethernet network (generally Ethernet).

2. **Double WiFi:** Similar to the above, the SBC uses two WiFi interfaces—one for ad hoc mode with the robot and another for internet access. The Remote Access client typically connects to the interface with internet access.

3. **Single WiFi:** This is used when the robot doesn't rely on WiFi (e.g., Arduino via USB) or its WiFi operates in Client mode, connecting to the local network. The SBC and Remote Access client connect to this local network.

Using the Remote Access Client

The Remote Access Service Client (RASC) runs on mobile devices like Android or Chromebooks. Ideally, this would be run on a device with a large screen, mouse, and physical keyboard, such as a Chromebook. This cost-effective configuration provides the resolution, screen size, and input devices most compatible with a Windows PC. Using a touchscreen-only device, such as a phone, could result in a poor experience for most tasks.

Server Selection Screen

The main screen lets you input the IP address, port, and password. Additionally, any remote access servers on your network will broadcast and appear on the list below. Selecting one still requires you to enter the password.

Press the **CONNECT** button to connect to the specified Remote Access Server.

Remote Control Screen

This screen mirrors the ARC PC's monitor. Clicking or touching the screen simulates mouse clicks on the ARC PC. On devices like Chromebooks, the mouse integrates seamlessly for intuitive use.

Audio Redirection

The Remote Access Server redirects audio between the client and the server. For example:

- The client device's microphone audio is sent to the ARC PC as its mic input in real time.
- All audio from the ARC PC's speaker is played through the client device's speaker.

Audio Redirection Setup Instructions

The redirection of Mic audio uses the virtual cable software VB-Cable. This application installs a driver to allow audio routing between inputs and outputs. This software is necessary for the RAS Client to stream mic data into the ARC PC. The screenshot below demonstrates a standard configuration of your Windows Sound Settings after completing the instructions.

1. The output device (#1) is configured for your speakers and *NOT* the "***CABLE (VB-Audio)***"
2. The input device (#2) is configured for "***CABLE Output (VB-Audio Virtual Cable)***"

1. Install the VB-Cable Virtual Audio Device Driver. [Download]
2. Right-click the speaker icon in the ARC PC taskbar to access sound settings.
3. Select the **Cable Output (VB-Cable Virtual Cable)** as the default input device.

   Note: Leave the output device set to the PC's default speaker.

4. To prevent sound duplication, mute the volume on the ARC PC.

Enabling Remote Access Server in ARC

1. From the ARC top menu, select the **Options** tab.
2. Click the **Preferences** button to open the preferences popup window.
3. Select the **Remote Access** tab to view the server settings.
4. Check the **Enable** box to activate the server.
5. Enter a memorable password.
6. Leave other values at their defaults until you are familiar with their functionality.
7. Click **OK** to save your settings.

Screen Scaling

There are two types of screen capturing methods built into the **RASS (Remote Access Server Service)**:

DirectX Mode

This is the fastest mode, which uses around 1% of CPU on an i7 and is ideal for robots with dedicated computers.
The display scaling must be set to **100%**, and this mode will be automatically used.

BitBlt Compatibility Mode

This mode is designed for computers that are not dedicated to a robot and have display scaling other than **100%**. Expect around 3% CPU usage on an i7 cpu. It uses slightly more CPU but ensures compatibility with different scaling settings.

Automatic Selection of Screen Capture Method

The **Remote Access Service (RAS)** will automatically choose the most appropriate screen capture method based on your display settings. Ideally, for the best performance, a remote-controlled computer should have its screen scaling set to **100%**. You can verify this in your display settings.

If this computer is not a dedicated robot PC, you may prefer to keep the scaling set to a higher value for better visibility. In this case, the **BitBlt compatibility mode** will be used automatically.

How to Check Your Screen Capture Mode

You can determine which screen capture method is being used by checking the **ARC Debug Log** when RASS starts. The log will display the detected scaling factor and the screen capture method in use.

Example: 100% Scaling (DirectX)

```
2025/01/30 17:30:16 -05:00     RAS: Scaling factor is 1%
2025/01/30 17:30:16 -05:00     RAS: Using DirectX for fast screen capture.
2025/01/30 17:30:16 -05:00     RAS: (raServer log) Waiting for client..
2025/01/30 17:30:16 -05:00     RAS: Broadcast Server: Started
```

Example: 125% Scaling (BitBlt Compatibility)

```
2025/01/30 17:30:16 -05:00     RAS: Scaling factor is 1.25%
2025/01/30 17:30:16 -05:00     RAS: Using BitBlt compatibility mode for screen capture.
2025/01/30 17:30:16 -05:00     RAS: (raServer log) Waiting for client..
2025/01/30 17:30:16 -05:00     RAS: Broadcast Server: Started
```

Setting Screen Scaling in Windows

To ensure optimal performance, follow these steps to adjust your screen scaling in Windows:

1. Right-click on the desktop and select **Display settings**.
2. Scroll down to the **Scale and layout** section.
3. Under **Change the size of text, apps, and other items**, select **100%** for the best performance.
4. If you are not using a dedicated robot PC, you may keep the scaling higher, and the system will automatically use BitBlt mode.

Diagnosing Remote Access Server

You can verify the server's status in the ARC Debug Log window. Messages will indicate the Remote Access Server's activity, including audits of your audio configuration to ensure the VB-Cable virtual device is installed and selected.

The example image above shows a successful configuration. The VB-Cable was discovered as the default input source, and RAS was started correctly.

Security Recommendation

Although the RAS Server is secured with a password, it is strongly advised **not** to expose the RAS Server port directly to the public internet. This application grants unrestricted

access to your Windows ARC PC, including control over its resources and functionality, and should, therefore, be handled with utmost caution. Exposing this server to the public internet significantly increases the risk of unauthorized access, which could compromise your system and sensitive data.

If you require remote access to your ARC PC, we strongly recommend implementing a secure Virtual Private Network (VPN) connection to the ARC PC. A VPN provides an encrypted, private tunnel for data transmission, significantly reducing the risk of external threats. Additionally, ensure that access to the RAS Server is limited to devices within your local private network. This configuration enhances security by preventing unauthorized connections from external networks. By following these best practices, you can ensure a robust and secure environment for accessing and managing your ARC PC.

Account

ARC requires a user account that was created on Synthiam.com website. The currently logged-in user is displayed in brackets of the account button text. Pressing the button will load the account settings menu.

Account Settings Menu

A new Synthiam ARC account can be entered here - or - the password can be changed for an existing account if it was previously changed on the synthiam.com website.

Shortcut Creator

The shortcut creator is a wizard for saving a shortcut to an EZB file. A script can also be configured to run when ARC loads with the specified file. By running the script automatically, you can execute connections to an EZB and initialization.

*Note: The current project will need to be saved before the shortcut creator will load. This is to ensure there is a valid ARC project file for the shortcut creator to use.

**Note: Windows must be configured to auto-login as the current user without a password for the shortcut creator to auto-load the project when Windows starts.

The shortcut creator can be found in the Options ribbon menu.

When the shortcut creator is loaded, there will be numerous steps to follow to define the shortcut options. Press the NEXT button after completing and verifying each step.

Step 1 - ARC Location

Verify the location of the ARC.EXE executable. This will be detected automatically, but it is worthwhile verifying.

Step 2 - ARC Project Location

The wizard will display the file location of the current project. The file displayed is the file that will load into ARC when the shortcut is executed. Verify the filename and path is correct

Step 3 - Auto-Start Script

When ARC loads the project, you can configure a scripting robot skill to execute automatically. Use this script to connect to the EZBs and initialize the robot. All of the scripting robot skills will be detected and listed in this drop-down. Select the script that you wish to auto-start from the drop-down.

Step 4 - Shortcut Hotkey

You can configure a key combination (hotkey) to load this project when the key is pressed automatically. This means you can hit a combination of keys at any time in Windows and the project will load with ARC.

Save Shortcut

Now that everything has been configured for the shortcut, you can save it. When the Save Shortcut File button is pressed, the default folder will be the Windows Startup folder. If you save the project in this folder, you will load the shortcut every time Windows starts. This is useful for embedded PCs within the robot or dedicated PCs for a robot.

Tips

The final step for the wizard is a list of tips to improve the performance of Windows for ARC and your robot. Pressing this button will open the web browser to a list of tips to get the most out of the computer. This includes information for configuring a dedicated embedded robot computer or improving performance.

If you would like the project to start with Windows, save the shortcut in the shell:startup folder. In the folder dropdown at the top of the Save Shortcut window, enter "shell:startup" and press <enter>. That will bring you to the startup folder of Windows where you can place the shortcut file. Next time Windows restarts, it will execute that shortcut file.

Servo Controls

This tutorial will provide technical information about servo motors and how they work. We made it easy to get a robot up and running. However, there are many fun and exciting things to learn about how the robot works. The more you know, the more you can get your robot to do it!

Types of Servos

ARC supports all servos, including the ability to define your servo driver using the Servo Script Robot Skill. The most common types of servos are...

- PWM Hobby servos that are connected directly to EZB digital ports
- Smart (Robotis Dynamixel, Lynx Motion, Feetech, Kondo KRS, LewanSoul, UB-Tech, and more)
- PWM servo extenders (SSC-32)

*Note: Servo driver robot skills are in the Servo Robot Skill Category. These skills are also listed at the bottom of this document.

Servo Ports (Vx and Dx)

There are two types of servo ports for ARC: those starting with the letter D and those starting with the letter V.

Dx servo ports range between 0-24 (D0-D24) and are directly connected to the EZB microcontroller firmware's communication protocol. This means that if you specify the position of a Dx servo port, the value will be sent to the EZB microcontroller using the communication method. Dx servo ports are, therefore, directly controlling an EZB microcontroller servo pin. Because of this, there may be a mapping between the Dx port in ARC and the physical EZB microcontroller pin. When selecting a servo port, the dialog will display the EZB and mapping pins. Reference the mapping pins of the EZB when selecting the Dx port in the dialog.

Vx servo ports are virtual servo ports ranging between 0-99 (v0-v99). They are used internally within the ARC framework and not connected to an EZB via the communication protocol or the firmware. Changing the value of a Vx port signals an event within the ARC framework for other robot skills to act upon. The best example is using a Robotis Dynamixel smart servo as you select the Vx port for the corresponding ID. When the Vx value changes, the Dynamixel robot skill will notice the new value and send it to the corresponding servo. The Vx virtual servos allow servos that have respective robot skills to control unique protocols. Another example of using a Vx servo port is a servo controller, such as the SSC-32 or Pololu Maestro.

Servo Interface Menu

ARC displays a standard configuration dialog for configuring servos across all robot skills. This manual will explain how to configure a servo (even multiple servos) to be moved from a robot skill. In this example, we will use a vertical servo robot skill, although this procedure applies to any robot skill that uses servos. Many robot skills use servos, including WiiMote, MYO, Camera, and dozens more...

The servo interface menu is standard across all robot skills using servos in their configuration menu. This menu will be displayed when selecting a servo in a robot skill's configuration. By default, you can choose the port and EZB for the servo, the MIN position value, and the MAX position value. You may also select the checkbox to invert the direction.

Advanced Servo Interface Menu

This new menu will display when the advanced button is pressed on the standard interface menu. This advanced menu allows the user to specify additional options and add multiple servos to an option. Press the Advanced button to add various servos or configure advanced settings.

***Note:** *Hover your mouse cursor over the blue question marks to read about advanced options. Many advanced options require hardware support to work. So, by default, they are set to -1, which means ignore.*

Advanced settings such as...

- **Min/Max**

  The minimum and maximum limits the servo can move. This is between 1 and the global max servo resolution value (default 180).

- **Ratio**

  The ratio is a multiplier of the position from the first servo. It is important to note that the ratio of the first servo will always be 1. This is because it is the master servo that all consecutive servos will reference for their position * ratio. Even if you attempt to change the first servo ratio, it will restore to a value of 1.

- **Velocity**

  If supported by the servo/driver/ezb, the velocity will control the servo's velocity. Check the manual for your specific servo to understand the value range. The values are not specific to ARC because they are unique to each servo.

- **Acceleration**

  If supported by the servo/driver/ezb, the acceleration will control the servo's acceleration. Check the manual for your specific servo to understand the value range. The values are not specific to ARC because they are unique to each servo.

- **Speed**

If supported by the servo/driver/ezb, the speed will control the servo's speed. Check the manual for your specific servo to understand the value range. The values are not specific to ARC because they are unique to each servo.

Specify Servo Resolution

By default, the software will have a servo resolution of 180 positions. While most hobby servo controllers (i.e., Arduino, EZ-Robot EZB) are limited to 180 degrees, some servos support a much higher resolution (i.e., Dynamixel). You can configure ARC to support a higher servo resolution per project. The global servo resolution setting can be configured in the My Robot Project Properties menu. The range will be calculated to compensate if the value of servo positions is higher than the EZB supports. For example, if you set the ARC servo resolution to 360 while using a PWM Servo, the value 360 will become the new maximum servo position. Still, the servo will not have any higher resolution because PWM servos are limited to 180 degrees. However, a dynamixel servo or stepper motor would use the 360 degrees. Remember, servos can only move in positive integer values.
***Note:** *The maximum servo resolution supported in ARC is 2,147,483,647*

Most Popular Servo Robot Skills

Many robot skills control servos, such as the Camera Device, WiiMote, Joystick, and scripting. Here are a few of the most popular servo skills used by robot builders to get started.

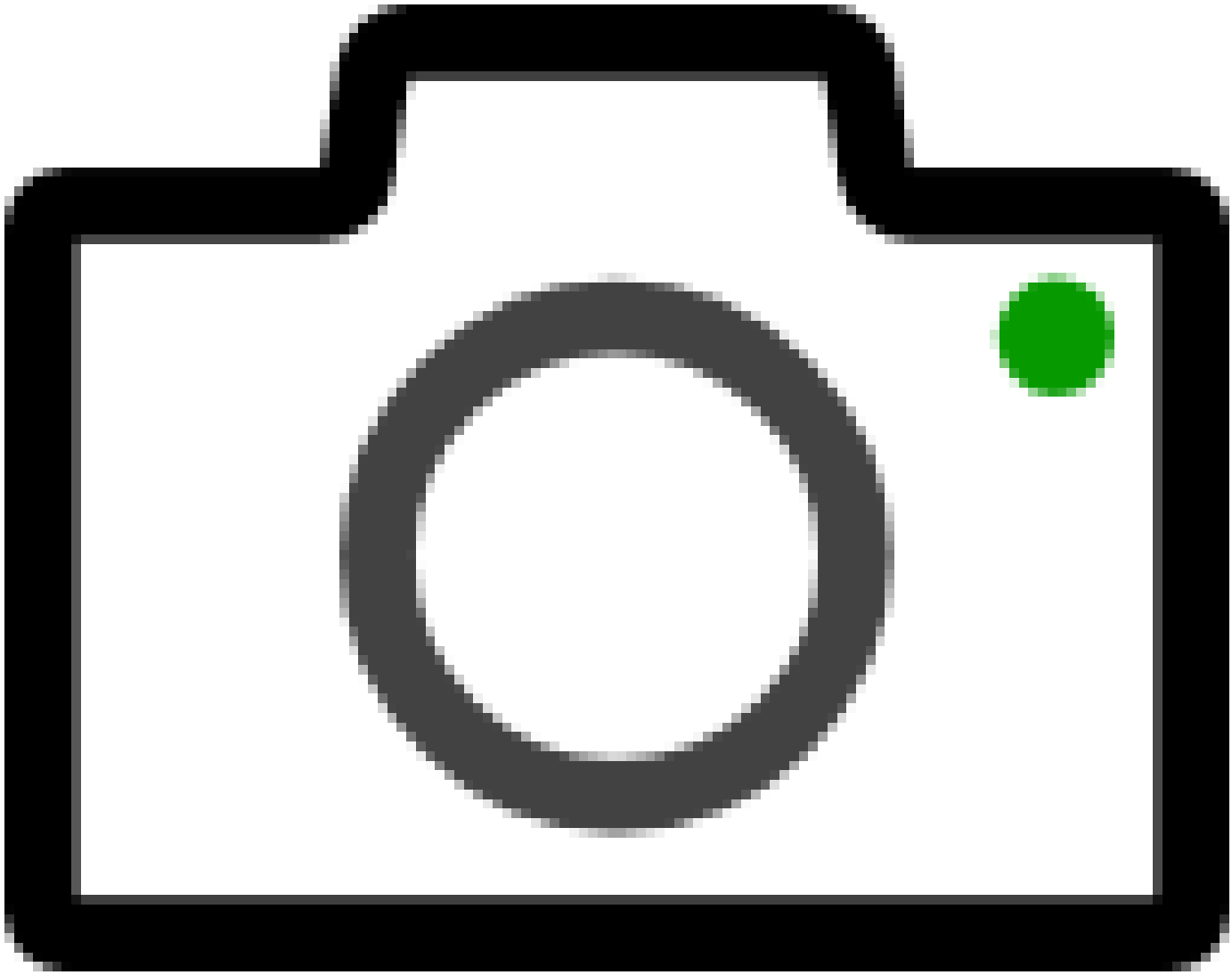
Vertical/Horizontal Servo

These servo skills allow you to drag the mouse horizontally (left & right) on display to move the position of a horizontal servo. The horizontal servo will only move to the minimum and maximum limits you specify in the skill's settings.
These robot skills are excellent for testing your robot's movement range and ensuring the servos work. We always recommend using these robot skills when building your robot to ensure everything works correctly. At the same time, the two versions of this skill differ in how the user interface is presented by dragging vertically or horizontally. This does not have to matter based on the orientation of your servo in the robot itself.
These two skills can be used for testing as it is a matter of preference.

Horizontal Robot Skill

Horizontal Robot Skill

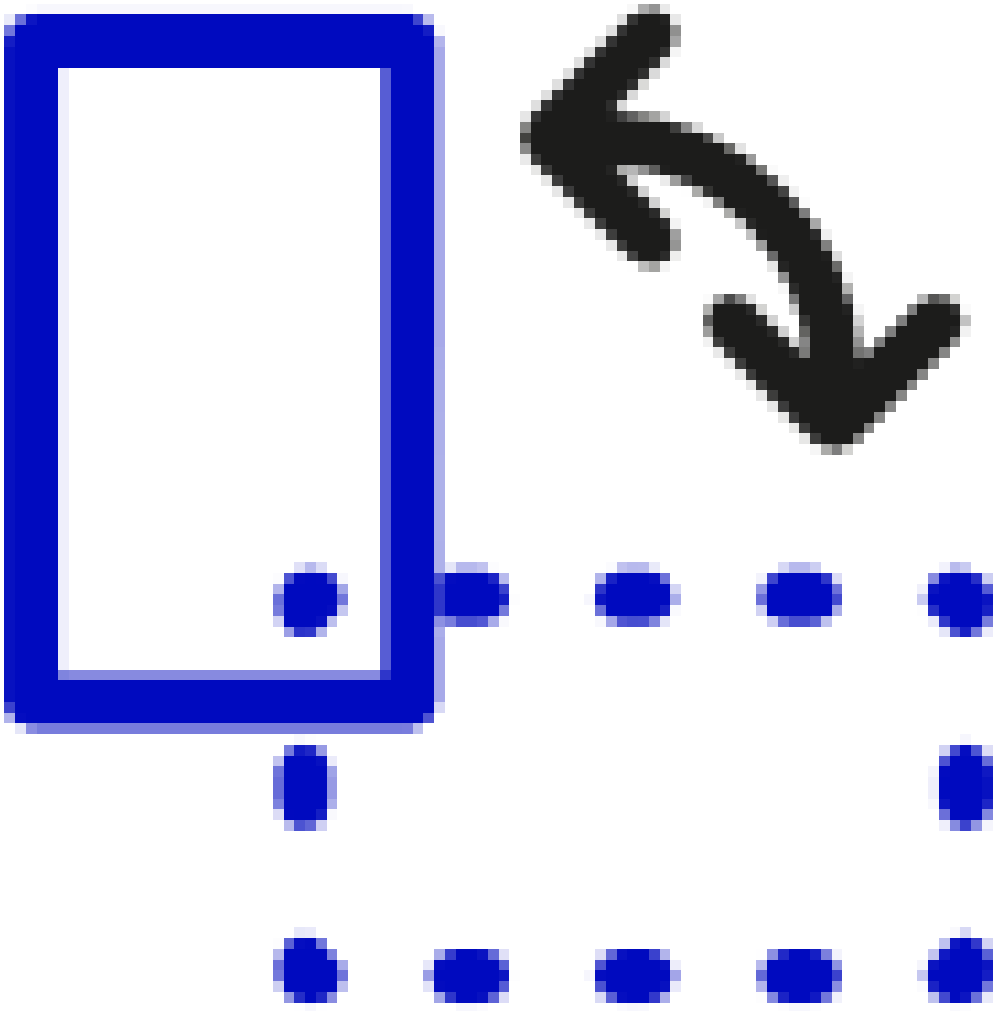
Camera Device



The camera device is one of the most potent and popular robot skills, demonstrating ARC's power.
This robot skill will use either the EZB remote camera (if supported) or a USB webcam mounted on the robot.
By configuring options, the camera can track objects, faces, and colors and even detect emotion or age with additional robot skills.
The Camera Device has settings in the options menu for controlling servos when a camera is mounted on a gimbal.
This feature allows the robot's camera to move and track the object.

Camera Device

Auto Position



Create servo animations to have your robot wave, dance, or pick up and move objects.
The Auto Position skill transforms a group of servos into custom positions (Frames) to create animations.
You can combine the frames into actions for your robot's animation.
This is done with "Inverse Kinematics" or "Motion Planning."

There are two types of the Auto Position robot skill. One is a movement panel that plays servo animations for robots that use servos to walk (i.e., humanoids or hexapods).
The other type is used for robots that use a different movement panel (i.e., hbridge, continuous rotation, etc.) but has servos for arms to be animated.

This is a popular robot skill for humanoid robots, such as the InMoov, Robotis Bioloid, or EZ-Robot JD/Six.
For robots with many servos used in arms or legs, this allows the creation of animations

that can be executed programmatically by other robot skills using the ControlCommand(). This means your robot can perform actions based on speech recognition or chatbot conversations.

Auto Position with movement panel

Auto Position for only servos

Joystick

Two types of joystick robot skills are based on the type of joystick used. The most common type is the XInput version, which supports the latest joysticks, including Xbox controllers. The xInput has additional support for analog trigger buttons and multiple independent joysticks. These robot skills can control servos, allow scripts to be assigned to buttons, and optionally control the current movement panel to move the robot.

Direct Input Joystick

xInput Joystick

Virtual Reality

Being able to control your robot servos with your hands and move the robot's camera with your head is a fantastic experience. A few robot skills support a variety of VR headsets, including the Meta Oculus Quest and Steam VR (i.e., HTC Vive, Windows Mixed Reality, etc.). The hand tracking feature is one of the great features of using the Meta Oculus Quest robot skill. This feature lets you move servos without holding the controllers in your hands. You can map your fingers to individual servos, which is excellent for humanoid projects like the InMoov.

Direct Input Joystick

xInput Joystick

Hard Set Servo Limits

With the appropriate Javascript, EZScript, or Python commands, you can set global servo positions across all robot skills.

The robot skill settings for servos are for the individual robot skill only. Every robot skill has servo values (i.e., min and max). This means the Camera skill has different servo values than the Joystick skill. The values specified in a skill's configuration are specific to that skill.

There are commands for all languages (JavaScript, Python, EZ-Script, etc.) for specifying limits. For example, the EZ-Script command in an INIT script to specify servo positions globally across the entire application is...

SetServoMin (servo port, position)

Set the minimum limit that this servo can ever move to

The servo position is between 1 and 180 (or global max servo value)

Example: SetServoMin(D14, 40)

SetServoMax (servoPort, position)

Set the maximum limit that this servo can ever move to

The servo position is between 1 and 180 (or global max servo value)

Example: SetServoMax(D14, 100)

Here's an example from the EZ-Robot JD project that prevents the left gripper from moving further in either direction globally across the ARC software. The appropriate commands for setting global servo position values are in the JavaScript, Python, or EZ-Script manual.

```
# Left Gripper
SetServoMin(d6, 30)
SetServoMax(d6, 90)
```

Example Testing a PWM Servo

Because ARC uses a standard dialog for configuring servos, the following steps will demonstrate how to use it. Any robot skill supporting moving servos will display the same "servo selection" dialog as this example.
In this example, we will use a standard PWM hobby servo connected to one of the EZB digital ports.

Step 1

Load ARC.

*Note: Always ensure you have the latest ARC. When you load ARC and connect to the internet, it will notify you of a newer version.

Step 2

Press the Project tab from the top menu. Now press the Add Control button.

Step 3

The Add Control window will display. This window allows you to browse and select controls to add to your project. Press the SERVO tab to view servo-specific robot skills.

Step 4

We are going to use the Vertical Servo control for this example. Many kinds of robot skills interact with servos, even more than you can see on this page. Nearly every skill control uses servos; however, only the specific servo skill controls are listed on this page. Even the Camera, WiiMote, MYO, and more use servos. Click the Vertical Servo button to add the vertical servo skill control to your project.

Step 5
You will add the Vertical Servo skill control to the workspace. This skill allows sliding the mouse vertically to move a servo position. Alternatively, a Horizontal Servo skill control enables the mouse to be dragged horizontally to move the position.

Step 6

As mentioned in the Controls Tutorial, every control has a gear button. You can press this gear button to load the configuration menu. Each control has a unique configuration menu.

Step 7
ARC will now display the configuration menu

Step 8

Each configuration robot skill will have many options. Any robot skill that uses Servos will have a similar servo configuration interface. Some robot skills may have two or more servo configuration interfaces (usually for horizontal and vertical servo robot skills). In this control, there is only one servo interface.

Name: This is the name of the robot skill.

Board Index: ARC can connect 5 EZ-Bs to the ARC Software. This specifies which EZ-B to send the servo command to.

Port: The EZ-B port of the servo. Pressing this button will display the EZ-B to select the respective port.

Min/Max: Min and Max limits in degrees of the servo. The servo can move between 1 and 180 degrees (or global max servo value). The value for Min must be less than the value of

Max in all cases, even when Invert is checked.

Invert: If the servo is moving in the wrong direction, checking this box will reverse/invert the direction of the servo.

Multi Servo: If more than one servo moves, you can specify multiple servos.

Step 9

Press the PORT button, and ARC will display the EZ-B port configuration. In this dialog, you may select the port and press the Close button.

Step 10

Now that you have selected a port, we can move the servo to specify the MIN and MAX limits. These numbers are in degrees between 1 and 180 (or global max servo value). Start with the MIN by pressing the mouse button while dragging the cursor UP or DOWN. The MIN value must be less than the MAX value. In this example, set the MIN to a low number, and the servo will move in real-time if connected to an ARC and an EZ-B.

Step 11

Move the MAX by pressing the mouse button while dragging the cursor UP or DOWN. The MIN value must be less than the MAX value. In this example, set the MAX to a low number, and the servo will move in real-time if connected to an ARC and an EZ-B.

Step 12
The Multi Servo button allows this control to move more than one servo simultaneously. Press the button to display the Multi Servo dialog.

Step 13

To add multiple servos, press the ADD SERVO button. Each time the button is pressed, a new servo entry is added. You can remove the servo by pressing the X on the respective servo. Use the PORT, MIN, and MAX to configure the limits. The multi-servo option also allows a ratio to be specified. The ratio is based on the primary value for the first servo in the list. If the servo degree position for the first servo is defined as ten and the ratio is 2, the respective servo will move to degree position 20.

Step 14

Close the servo control configuration dialog and return to the workspace. Now that the servo

has been configured, you may move it between your specified MIN and MAX limits. Click in the servo position, and the cursor will change while you hold the mouse button. Slide the mouse up and down (for vertical control) to move the servo between the specified limits.

*Note: Even though many controls may move servos, the configuration of those servos is only valid for that control. If you configure a camera control to move servos to specific min/max ranges, those ranges only apply to the camera control. Use the Relative Servo control if you wish a servo or group of servos to move together from the script.

Servo Robot Skill Drivers

While hundreds of robot skills use servos, that may not be their primary focus, and they are in different categories.
For example, the Camera Device is a popular robot skill, but its primary focus is the camera and tracking, so it is located in the Camera category.
The Servo category generally lists robot skills that are driver- or servo-specific. You will find the robot skill driver in the Servo category if you use an intelligent servo, such as the Robotis Dynamnixel.
Below is a list of all robot skills from the Servo robot skill category.

[opt:skillcategory:servo]

Numeric Slider

An Innovative Numeric Input Solution for Precision and Efficiency

The Number Selector as it appears throughout Synthiam ARC.

What the ARC Number Selector Does

The ARC Number Selector is the compact numeric control used throughout Synthiam ARC whenever a numeric value is required.

It looks simple but provides multiple convenient input methods for efficient and precise editing.

Common uses

- Servo positions in **Auto Position** frames.
- Servo values, limits, and other parameters in robot skill configuration windows.
- Camera device parameters (tracking thresholds, region sizes, speeds, etc.).
- Any numeric field across ARC that requires entering or adjusting numbers.

Beyond direct typing, the control supports dragging to change values, entering math expressions, and referencing ARC global variables such as `$myVariable`.

Anatomy of the Number Selector

The visible parts are simple and consistent:

- **Numeric label** – displays the current value; this area is what you typically click and drag.
- **Drag icon** – a small icon beside the number that indicates whether dragging is vertical or horizontal for adjustment.

The drag direction (vertical or horizontal) depends on where the control is used, but the behavior is the same: drag one way to increase and the opposite way to decrease the value.

Tip: When the cursor changes to a move cursor (↕ for vertical or ↔ for horizontal), you can left-click and drag to change the value.

Changing Values by Dragging

Quick steps to adjust a value by dragging:

1. Hover over the numeric label or drag icon until the cursor changes to the appropriate move cursor.
2. **Left-click and hold** on the number.
3. **Drag** in the indicated direction:

   Vertical control: drag up to increase and down to decrease.
   Horizontal control: drag right to increase and left to decrease.

4. Release the mouse button when you reach the desired value.

The increment step (for example, 1, 5, or 10) depends on the control's internal configuration for that context.

Fine vs. Normal Adjustment

Hold **Shift** while dragging to switch to an alternate sensitivity (often finer) for precise adjustments.

ARC uses configured sensitivity settings, so the exact change rate may vary by control.

Visual Feedback

- While dragging, the number's background may change (for example, to a light green) to show the control is active.
- If the value reaches zero or below, the text may turn **red** to indicate a special or potentially critical state (for example, a 0 servo position).

Typing Values, Expressions, and Variables

The Number Selector supports direct numeric entry, simple math expressions, and references to ARC global variables. This allows quick calculations and dynamic values without leaving the control.

Opening manual entry

1. **Right-click** on the numeric label.
2. The numeric label and drag icon will be replaced by a textbox for manual input.
3. Type a number, expression, or a variable expression and press Enter to apply.

Keyboard shortcuts:

- Enter – evaluate and apply the entry.
- Esc – cancel manual entry and restore the prior value.

Plain numeric entry

Type a simple number, for example:

```
90
180
45
```

On Enter, the value is applied and clamped to the control's defined Minimum and Maximum.

Math expressions

Enter basic mathematical expressions to compute values on the fly. Examples:

```
90 + 10
(180 / 2)
45 * 3
(10 + 5) * 2
```

ARC evaluates the expression when you press Enter and uses the resulting number as the control value.

Using ARC global variables

You can reference ARC global variables (the same ones used in scripts and other controls) by prefixing with `$`, for example `$myServoPosition` or `$panCenter`.

```
$myServoPosition
$panCenter + 10
($tiltMax - $tiltMin) / 2
$cameraTrackingThreshold * 0.8
```

- The referenced global variable must exist and evaluate to a numeric value.
- After evaluation, the result will be clamped to the control's Minimum and Maximum.

Minimum, Maximum, and Clamping

Every Number Selector has defined **Minimum** and **Maximum** limits that depend on its context (for example, servo positions often range from `1` to `180`).

When you drag or enter a value (or expression), ARC evaluates the result and then clamps it:

- If the result is less than Minimum, the value is set to Minimum.
- If the result is greater than Maximum, the value is set to Maximum.

Note: Clamping protects your robot and configuration from invalid or unsafe values, particularly for servos and motors.

Where You Will See This Control in ARC

The Number Selector is a shared control used widely across ARC. Typical locations include:

- **Auto Position** – defining servo positions in frames and fine-tuning poses.
- **Robot skill configuration windows** – setting speeds, delays, thresholds, and numerical limits.
- **Camera Device** – configuring tracking types (color, object, face), region sizes, sensitivity, and related parameters.
- **Other settings** – timers, counters, and various numeric fields in plugins and general configuration panels.

Example: Number Selector used in a configuration window for parameter adjustments.

Once you are comfortable with dragging, typing expressions, and using ARC variables, the workflow is the same everywhere the Number Selector appears — providing a consistent numeric input experience across Synthiam ARC.

Quick Reference

Action	Result
Left-click + drag	Adjust value (direction and speed depend on drag direction and sensitivity).
Hold Shift while dragging	Use alternate (often finer) drag sensitivity.
Right-click on value	Open manual entry textbox for typing numbers, expressions, or variables.
Type a number (e.g., `90`)	Set exact value (clamped to Min/Max on Enter).
Type an expression (e.g., `90 + 10`)	Evaluate the expression and use the numeric result.
Use global variable (e.g., `$myVariable + 5`)	Evaluate using the ARC global variable's numeric value.
Enter in manual entry	Apply the expression, clamp to Min/Max, and close the textbox.
Esc in manual entry	Cancel and restore the previous value.
Value becomes red	Indicates the value is zero or below — a special or warning state.

Port Selection (servo/digital/pwm)

ARC presents a standard window for selecting digital, serial, servo, or PWM ports. This

window also displays the type of EZB that ARC is currently connected to. If there is no connection to an EZB, ARC will display a default image of an unknown EZB.

Unknown EZB (no connection)

Known EZB (connected)

Preview EZB

If there is no EZB connected, the default EZB image is displayed for an unknown EZB. To preview any EZB and its connections, use the dropdown to select the EZB from the list.

EZB Manual

Every EZB has a manual page with information about the firmware, pinout mapping, installation, drivers, etc... To view the manual page directly from ARC, select the appropriate EZB and press the View EZB Manual button. You will require an internet connection for this feature. This feature is specifically essential if for EZBs with multiple firmware options and pin mappings.

Where's This Window?

The port selection window is displayed whenever a servo, PWM, serial, or digital port is selected. These options are available when the configuration button of a robot skill is pressed. When configuring a robot skill, ARC may present you with settings to configure a port. Pressing the port button will display this port selection window. This example image is the configuration menu for a Vertical Servo robot skill.

Movement Panels

A Movement Panel is a type of robot skill responsible for moving the robot in directions (forward, left, right, stop, reverse, etc.). A movement panel robot skill integrates with the ARC environment and allows other controls or scripts to instruct the robot to move in any direction. When a movement panel is added to your project, it registers itself as the responsible control for directional movements. ARC has many movement panels. For example, there are HBridges, Continuous Rotation Servos, Auto Position GAITs, AR Parrot Drones, and iRobot Roombas.

An advantage of a Movement Panel is that you do not need to code logic for the robot hardware to move. Instead, use movement commands with the optional speed parameter to have the movement panel respond automatically. That will instruct the Movement Panel to do its thing, even for an H-bridge, drone, gait, etc...

The advantage of using movement panels is agnostic robot hardware per project. Consider if you built a robot that performs a specific task using an H-bridge movement panel. You can replace the hbridge Movement Panel with any other movement panel (such

as a drone or Auto Position for Hexapod or humanoid).

Any project can interchange any movement panel with any other movement panel. The advantage of using movement panels is not to require code for a specific hardware configuration, thus allowing agnostic hardware configurations per project.

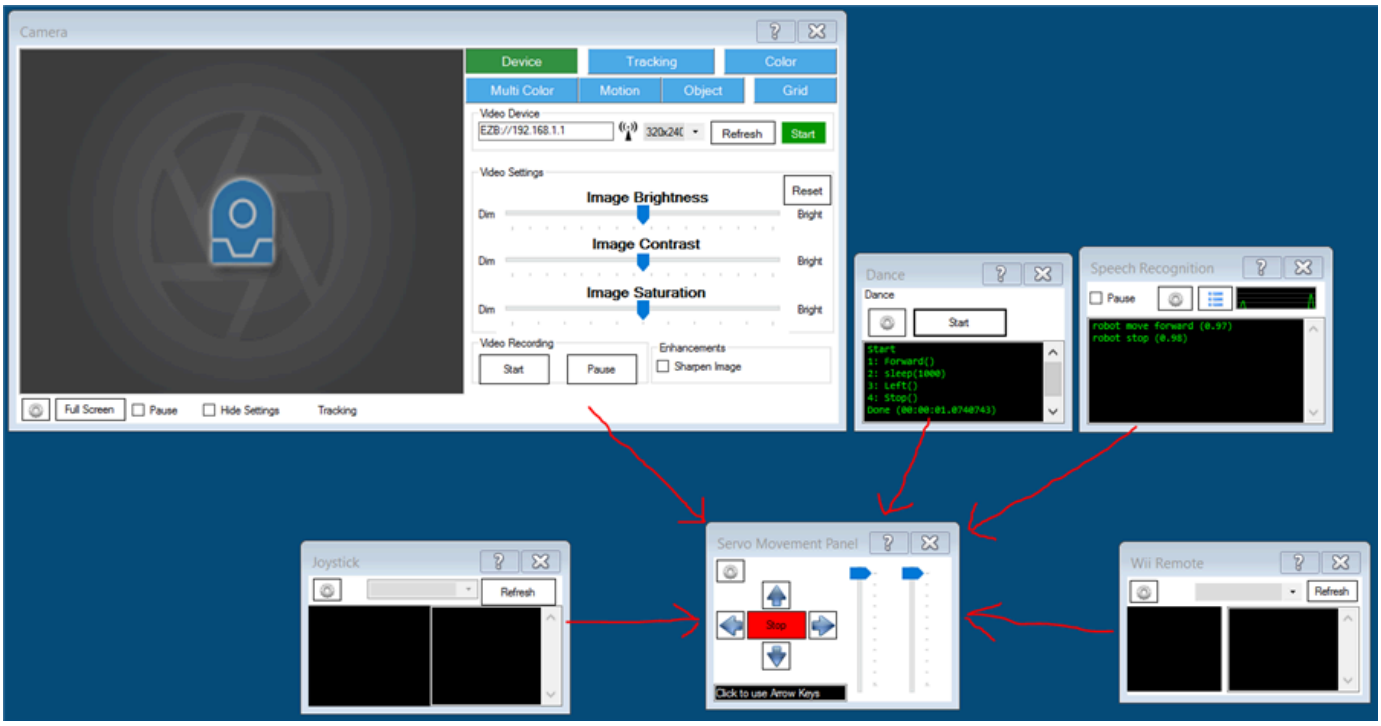
***Note:** You may add a maximum of one movement panel to a project, AND the movement panel generally is on the first ezb (connection 0). See the *Getting Started* section to choose an appropriate movement panel for your custom robot.

Contents

- Project With Movement Panel
- Directions
- Speed
- Scripting
- Speech Recognition
- Other Robot Skills
- Camera Control
- User Interface Builder
- Programming and Variables
- RoboScratch
- Blockly
- Create Custom Movement Panel

Project With Movement Panel

This screenshot is of a robot project that contains a movement panel. All other robot skills can send commands to the movement panel to control the robot. For example, with a checkbox in the camera device, the robot can follow a ball for any movement panel and any robot type. The same code can be used for a different robot hardware configuration by swapping the movement panel with another.



Directions

The ARC framework provides several pre-programmed directions: forward, left, right, reverse, stop, roll right, roll left, up, down, and custom. Movement panels will support the directions that are appropriate to their movement type. For example, an HBridge movement panel will not support UP or DOWN because the robot must have wheels and cannot fly. You can see what directions are supported on a movement panel by the buttons visible on the interface. This diagram demonstrates how the various directions are expected to behave by movement panels. For example, turning left or right will likely rotate on the spot. However, turning slightly right while moving forward will require the robot to move forward with the right wheel speed somewhat slower than the left.

Speed

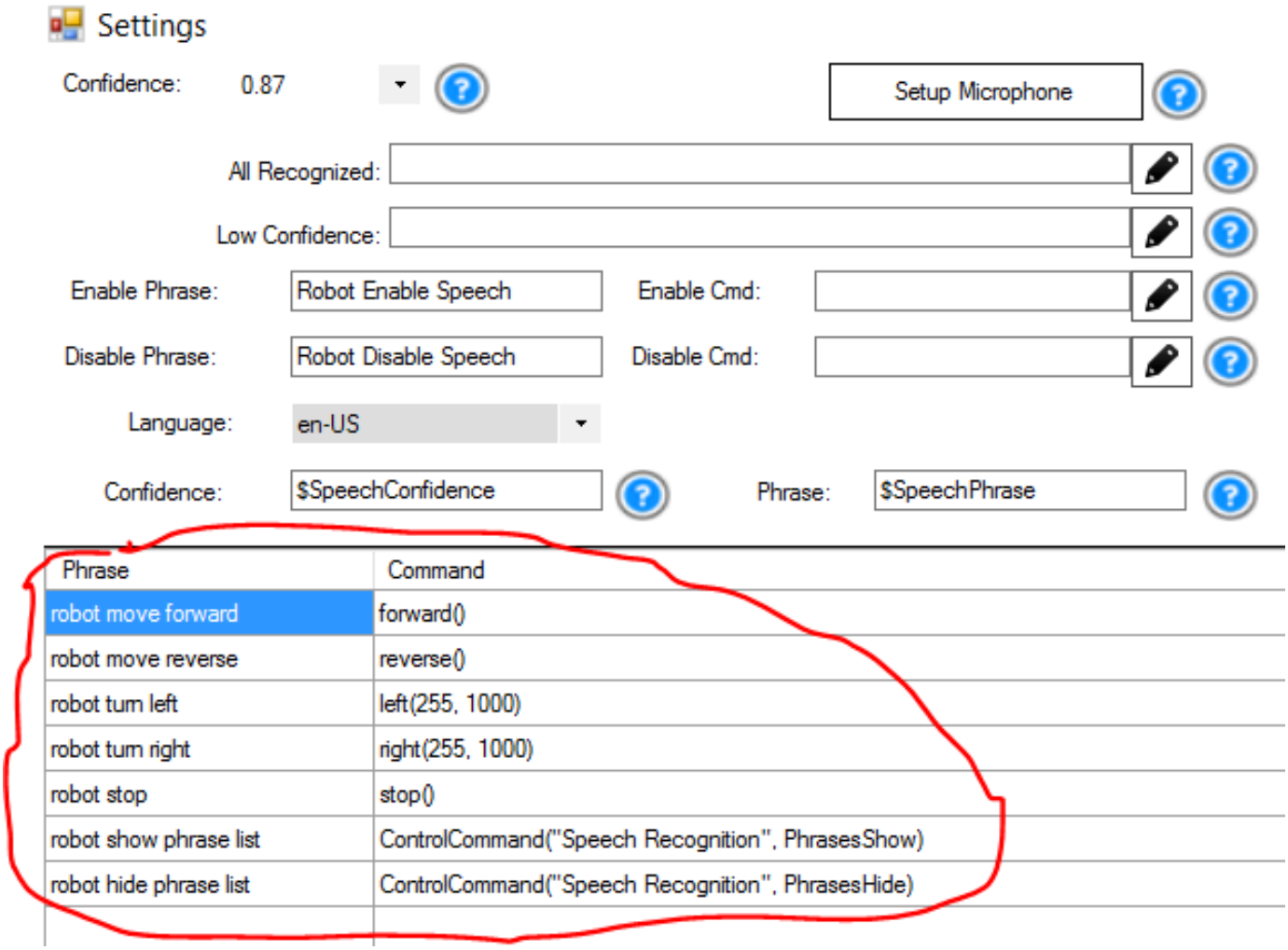
When specifying a direction to move, most movement panels will support a speed range between 0-255. The left and right wheels can have different speeds for driving forward with slight arc turns in either direction. This allows joysticks and tracking skills to control the robot's movement better.

Scripting

When a Movement Panel is added and configured in your project, documented scripting commands will instruct the movement panel to begin moving the robot.
If, anywhere in your project, you executed the script command Forward(), the script engine will instruct the project's Movement Panel to begin moving forward.
It is recommended to reference the script manually in ARC for the desired language. Click HERE for the manual of your preferred scripting language.

Speech Recognition

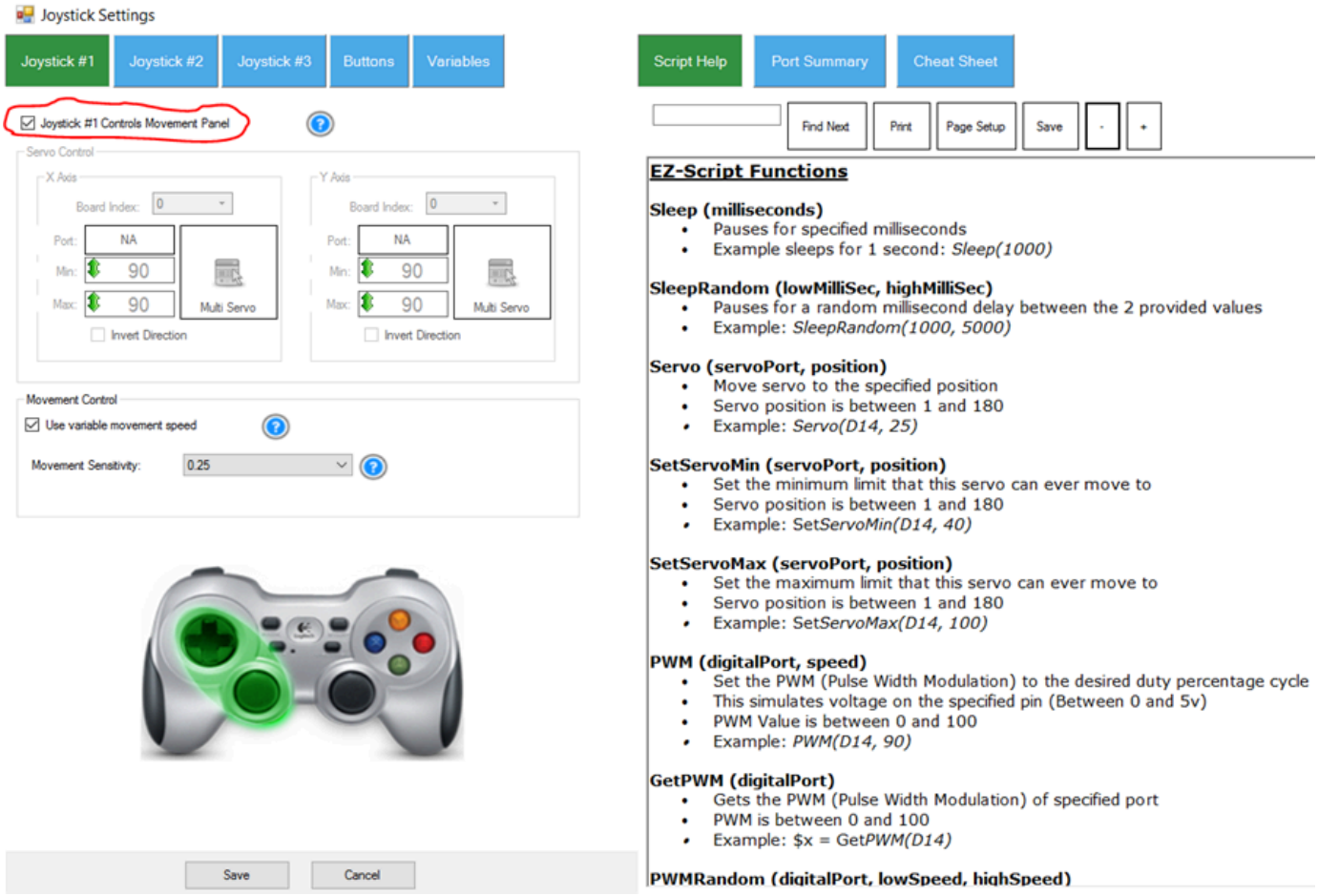
Some controls, such as the Speech Recognition Control, will trigger scripts based on user input (speech). The default configuration of the Speech Recognition Control includes speech commands and respective scripts for instructing the Movement Panel to move the robot. Viewing the Speech Recognition configuration, you will see script direction commands...



Phrase	Command
robot move forward	forward()
robot move reverse	reverse()
robot turn left	left(255, 1000)
robot turn right	right(255, 1000)
robot stop	stop()
robot show phrase list	ControlCommand("Speech Recognition", PhrasesShow)
robot hide phrase list	ControlCommand("Speech Recognition", PhrasesHide)

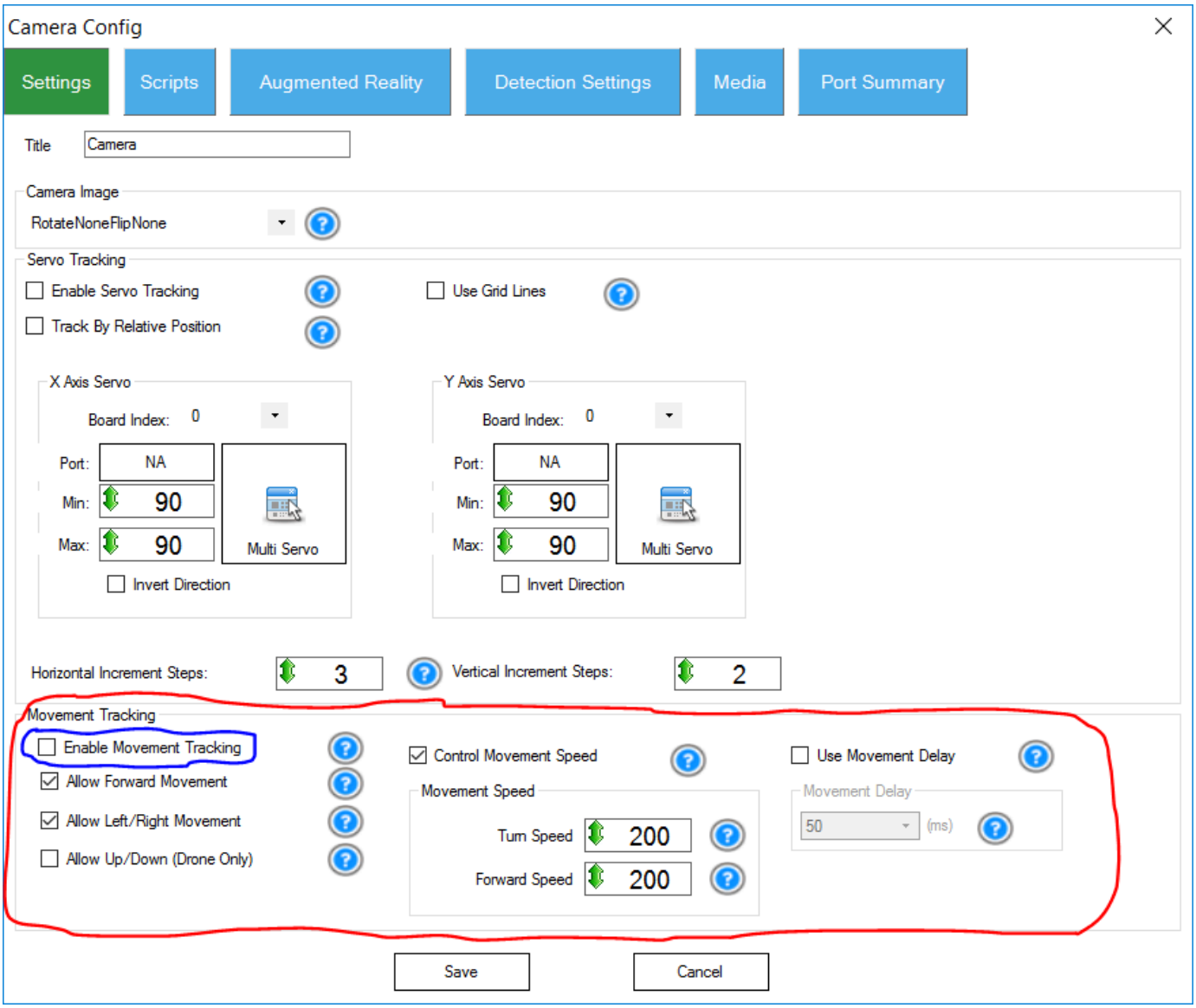
Other Robot Skills

Good question! Now that we know how the script can instruct a movement panel to move, how do other controls work with movement panels? Well, that's the magic of ARC. For example, if a Joystick Control was added to your project, pushing forward on the joystick will instruct the Movement Panel to begin moving forward. The Joystick control is pre-configured to send the Forward command to ARC. The registered Movement Panel will respond to the direction request. Let us take a look at the settings menu for Joystick Control. Here you will see the checkbox which assigns Joystick #1 to the "Control Movement Panel."



Camera Control

Yes! The camera control has an option in the settings to instruct Movement Panels to move in any direction based on the tracking method configured. The tracking method is how the robot will respond to tracking a specific tracking type. You can learn more about camera control and terminology by clicking HERE. If the Camera Control is configured to follow an object with MOVEMENT, it will instruct the current Movement Panel. Let's take a look at the Camera Settings and where the option is to have the camera control a movement panel.

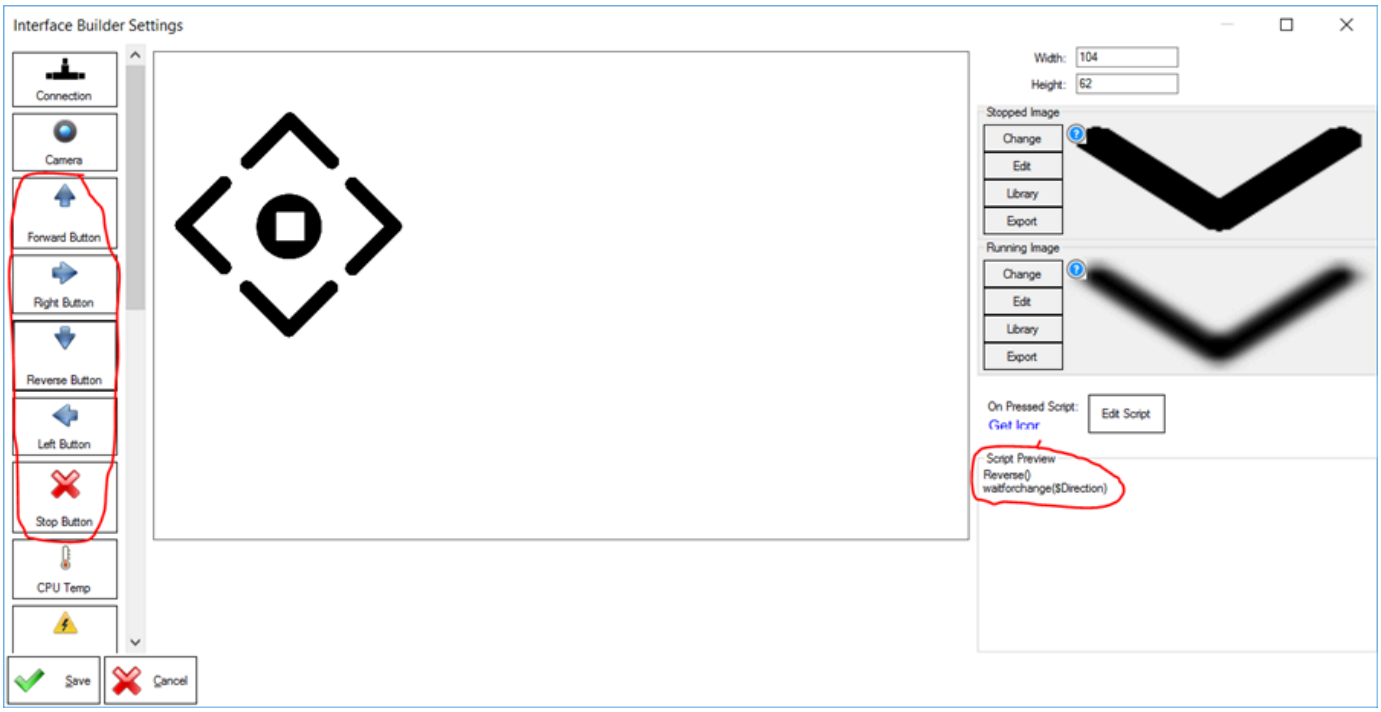


***Note:** Highlighted in RED is the section that contains options to configure how the Camera Control will communicate with the Movement Panel. Highlighted in BLUE is the checkbox which turns on/off the ability of the Camera Control to track movement. There are blue question marks that provide more information.

User Interface Builder

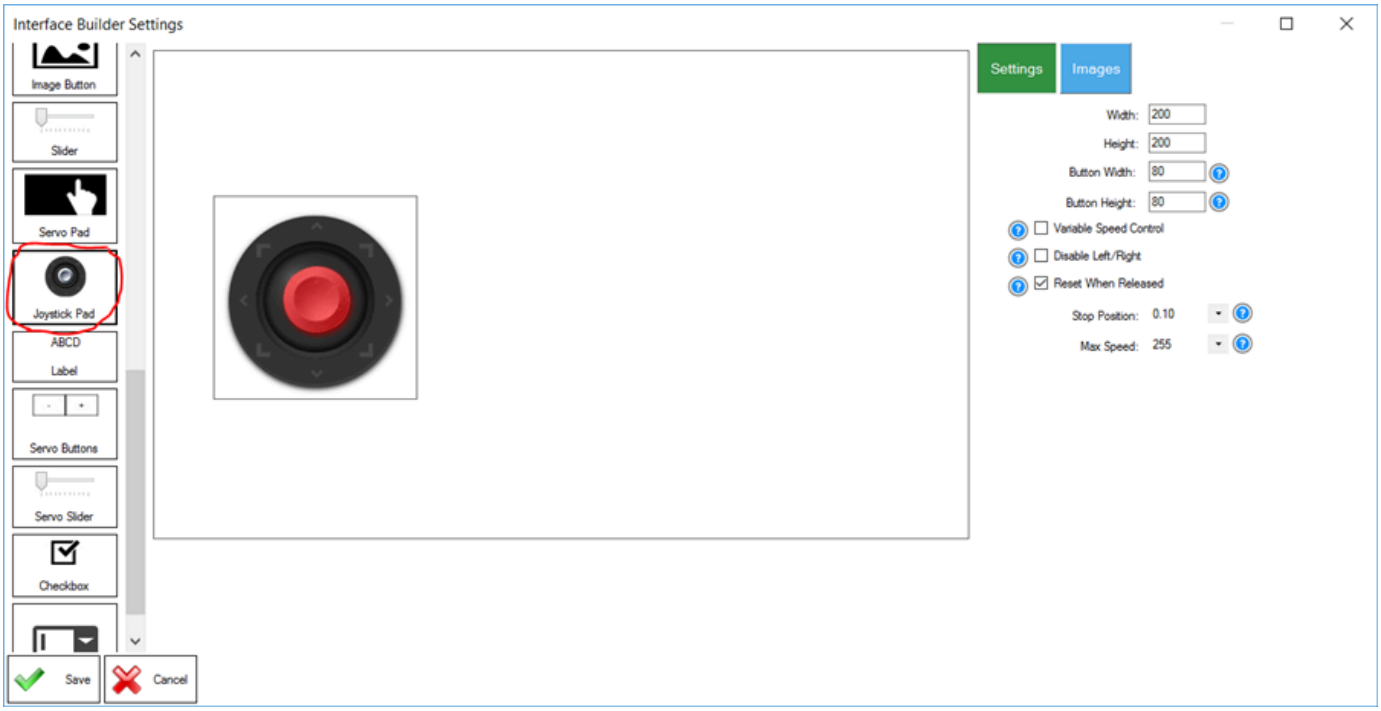
The trend in this lesson is that any control or script dealing with movement will instruct the current Movement Panel to move the robot in the specified direction. This also applies to widgets within the User Interface Builder. Within the User Interface Builder, you may add buttons to control movement direction with script commands (Forward(), Left(), Stop(), etc.). Or, you may add the Joystick Pad.

Here is an example of buttons added to the User Interface Builder, which will instruct movement using script commands (Forward(), Left(), Stop(), Right(), Reverse(), etc.).



Here is an example of using the Joystick Pad in the User Interface Builder, which automatically instructs the Movement Panel. One advantage to the user interface builder's Joystick Pad is that it can control the speed of the movement panel supports it.

*Note: You can tell if a movement panel supports speed because it has a speed slider.



Programming and Variables

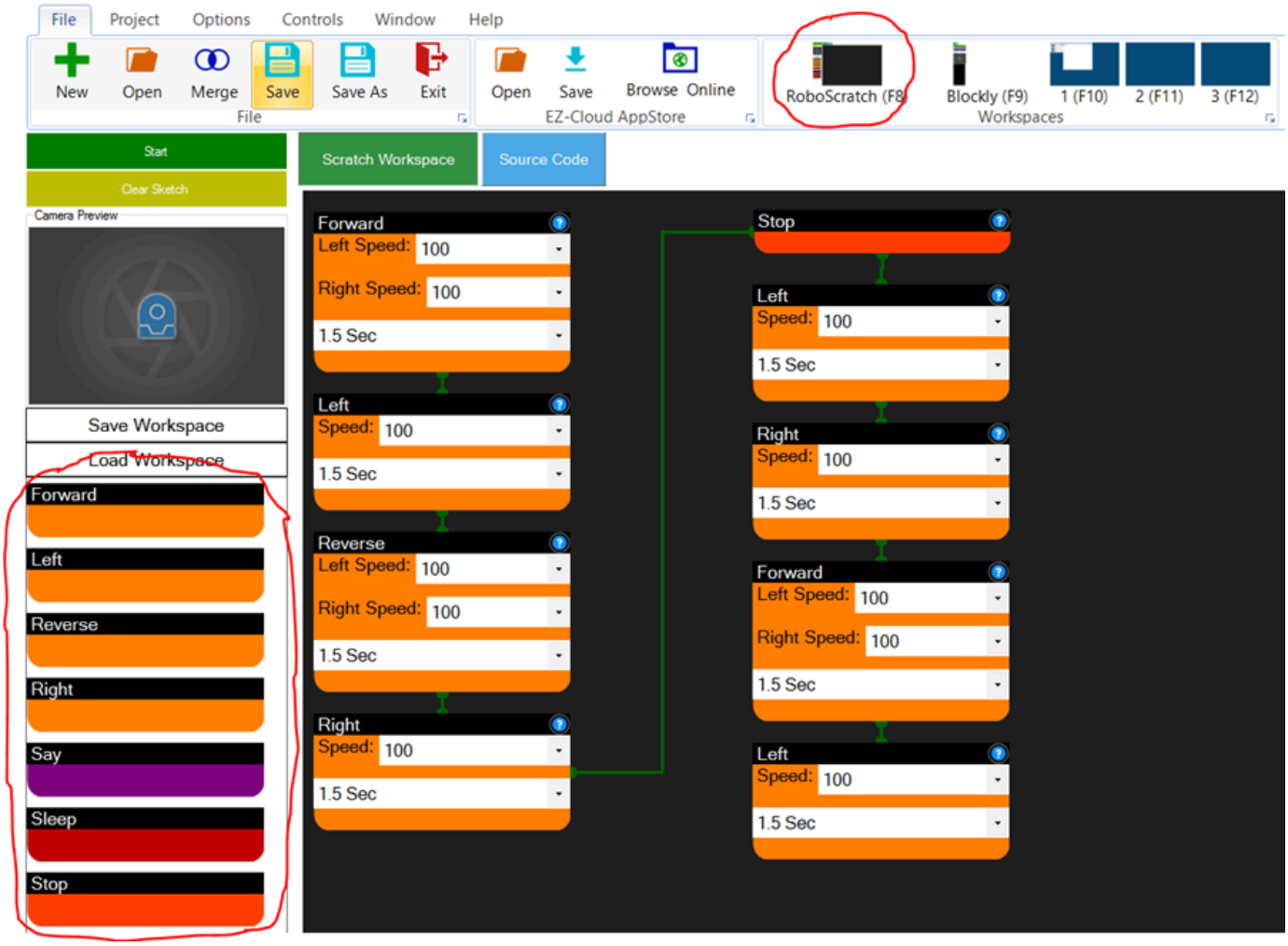
The current direction and speed of the robot's movement are accessible in scripting as global variables. Custom scripts can use these variables to know the direction or speed of movement. Or the variables are also used when creating a custom movement panel to understand the user-selected direction and speed. You can see the global variables if you add one of the variable watcher robot skills. The global $DIRECTION variable can determine the current direction. To choose how to access global variables, check the manual for the programming language that you're using. ARC can use different programming languages and access the global variable storage. For example, if using JavaScript, you can get the current direction with the getVar() command.

The current speed can also be obtained using the respective GetSpeed commands for your selected programming language. For example, if using JavaScript, you can get the left wheel speed with the GetSpeedLeft() command.

Now, we've covered getting the data from the movement, but you can also set the direction and speed of movement. This can be done using the movement commands and setSpeed commands. Again, you must check the manual for the specific programming language you're using with ARC.

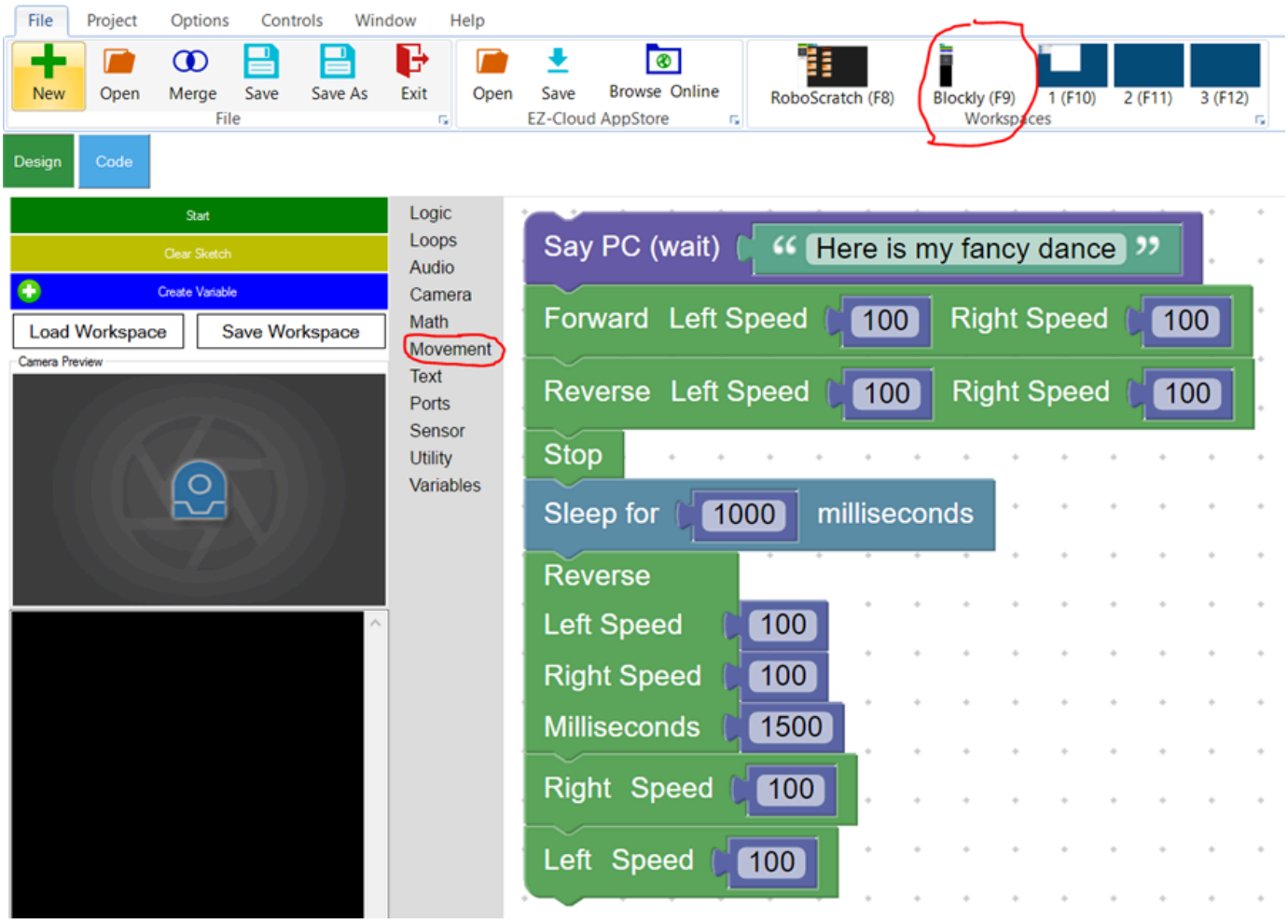
RoboScratch

Any reference to moving (Forward, Left, Right, Stop, etc.) will automatically instruct the project's current Movement Panel to begin moving. This includes using either RoboScratch, an excellent beginning programming interface with ARC.



Blockly

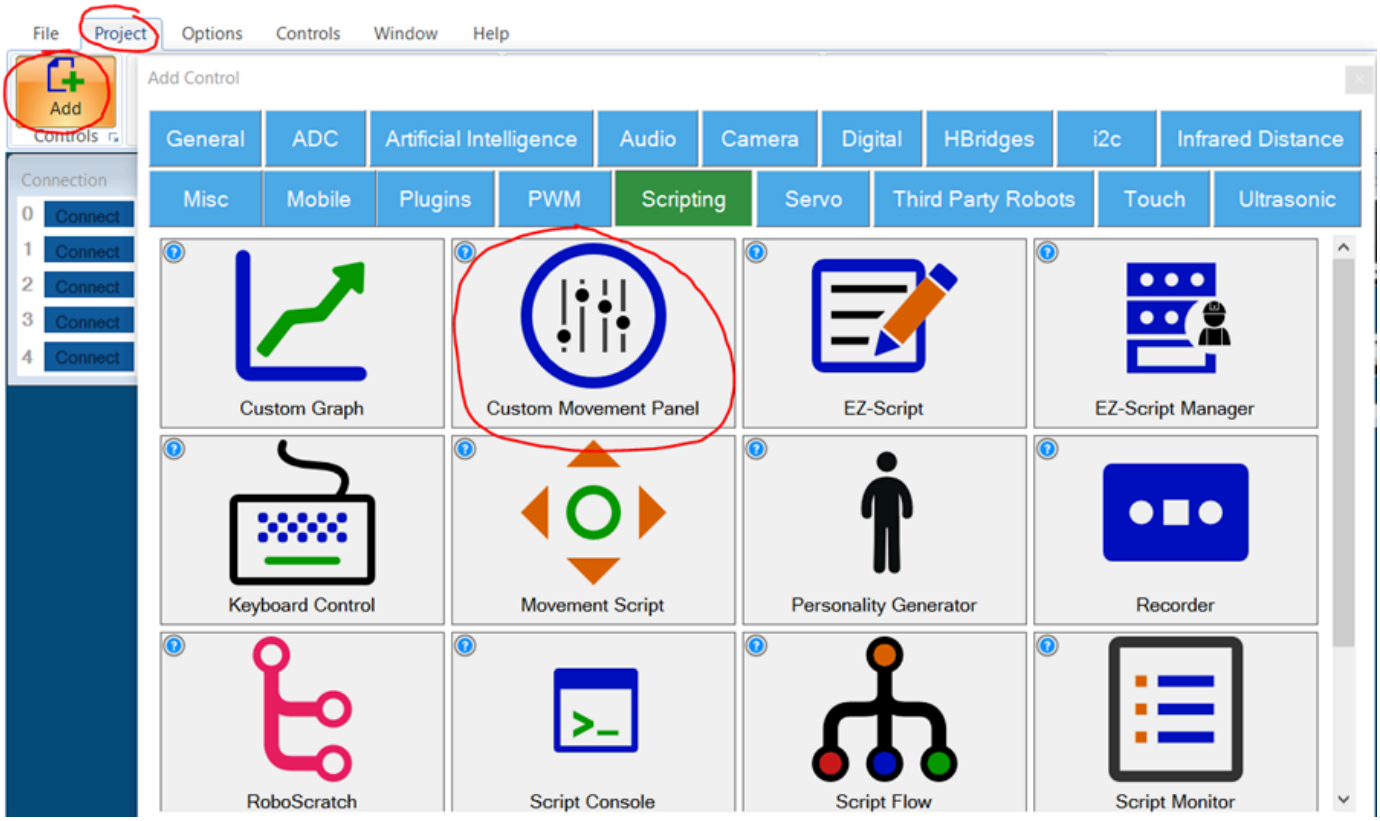
Any reference to moving (Forward, Left, Right, Stop, etc.) will automatically instruct the project's current Movement Panel to begin moving. This also includes using Blockly, an excellent intermediate programming interface with ARC.



Create Custom Movement Panel

As mentioned earlier, ARC has many movement panels for HBridges, Continuous Rotation Servos, Auto Position GAITs, and even AR Parrot Drones or iRobot Roombas. However, if you seek to create a custom movement panel, the Custom Movement Panel is what you are looking for. The Custom Movement panel contains events for each direction (Forward, Left, Right, Stop, Reverse) for which you may add custom code.

ARC will execute the event in the Custom Movement Panel when another control requests to move your robot (i.e., forward, left, right, stop, reverse). Yes, the joystick will control your custom movement panel. Yes, the camera will send commands to your custom movement panel. The point of this document is that any mention of a movement in ARC will instruct the movement panel to move the robot, even a custom movement panel.



All Movement Panel Robot Skills

[opt:skillcategory:movement panels]

Navigation Messaging System

The Navigation Messaging System (NMS) is built into the ARC framework. It allows skills to push data (location & obstacle detection) into the messaging system. Then, mapping/ navigation SLAM skills can subscribe to the navigation events.
The NMS was designed to provide transparent Navigation to the user in ARC. You do not have to write any complicated code because the robot skills will work for navigation and obstacle avoidance.

NMS Stack

The NMS stack consists of sensors that feed the messaging system, received by mapping skills.

Level #1 - Map & Navigation

This is how to view the sensor data and organize it into maps for navigating. There are a few skills that do this. They subscribe to the sensor event data and assemble a map. Then, they allow you to click on the map and instruct the robot where to navigate to (either way-point or any destination, depending on your sensor data). A few navigation skills are...

- [opt:skill:20956] (preferred)
- [opt:skill:20072]

- [opt:skill:15944]

Level #3 - Sensor Groups

You require sensors that feed data into the messaging system that maps are made from. The navigation messaging system collects data by supporting two sensor input groups...

1.
Lidar/depth scanners: any sensor that can detect obstacles (laser, 360-degree lidar ultrasonic, IR, etc.). The sensor will detect objects and mark on the map where the robot cannot go (i.e., walls, sofa, chairs, etc.). Supported skills are...

[opt:skill:20404]
[opt:skill:20086]
[opt:skill:16090]
[opt:skill:20127]
[opt:skill:20980]
JavaScript & Python commands for manually reporting obstacle detection

2.

Localization telemetry pose navigation:
Special sensors that keep track of the robot's position (pose) in the real world (IPS, wheel encoders, Roomba movement panel, intel realsense t265, etc.). *Note: only one Group #2 sensor can be active at one time.
[opt:skill:20067]
[opt:skill:17492]
[opt:skill:17591]
[opt:skill:19164]
[opt:skill:20456]
[opt:skill:21080]
JavaScript & Python commands for reporting positioning manually

What Sensors Do You Need?

Depending on what NMS Level 1 mapping/navigation you will be using, the sensor requirements may differ. For example, using the mapping skill [opt:skill:20956] will require a sensor from each group L3G1 and L3G2 (layer 3 group 1& layer 3 group 2). So check the mapping robot skill so you know what NMS layer 3 group sensors are required for it.

Scripting Support

NMS commands are supported by the JavaScript and Python scripting compilers in ARC. The namespace is *Navigation,* and You can view more details in the Support section for JavaScript or Python. The script commands allow custom sending Level #3 Group #1/#2 sensor data into the NMS. As well as pausing and stopping any ongoing navigation by a Level #1 navigator.

Cartesian Coordinate System

This robot skill uses cartesian coordinates to reference the robot's starting position. The starting position is always 0,0 and is defined at startup. As the robot navigates, the skill measures the distance from the starting position. The unit of measurement is in CM (centimeters). Read more about the cartesian coordinate system on Wikipedia.

Further Reading

Many experts have researched and documented robotics's navigation, localization, and known pose positioning. While the Intel T265 has demonstrated the most significant success at inside-out localization, many other methods have been attempted over the years. The topic of reach into robot navigation is far too vast to be covered in this manual. However, we'll leave you with a few papers that have greatly influenced the robot industry in this field.

- Where Am I? - Systems and Methods for Mobile Robot Positioning by J. Borenstein, H. R. Everett, and L. Feng
- Carleton Winter 2012 Position Estimation presentation

All Navigation Robot Skills

These are navigation robot skills that work with the NMS or assist with controlling the robot's movement.

[opt:skillcategory:navigation]

Virtual Workspaces

This feature provides additional virtual workspaces to organize robot skills.
The option is found on the File tab of the top menu. To organize robot skills, give workspaces names based on the robot skill type. Add audio-related robot skills to a workspace named Audio. The same applies to Vision, Movement, and all other robot skills.

Workspace Types

There are several default workspace pages that you can select from in the top menu bar. They are as follows...

- **Debug Log -** displays the ARC logs. It will display a white exclamation mark when new log data is available to view.
- **Full Screen Interface -** displays any Interface Builder screen in full-screen mode. Use this as App Mode.
- **RoboScratch -** a graphical programming environment for beginners.
- **Blockly -** a block-based programming environment for intermediate.
- **Virtual Desktop Workspaces -** the workspaces that contain your robot skills for the project.

Changing Workspaces

Locate the workspace section in the File tab of the ARC top menu.

Adding Workspaces

Press the Add button in the workspace section of the File tab.

Removing Workspaces

Press the Remove button in the workspace section of the File tab. The last workspace will be removed if it is empty of robot skills.
A message will be displayed if the workspace is not empty. Only empty workspaces can be removed.

Renaming Workspaces

Right-click a workspace and select Rename. Enter a name for the workspace.

Moving Robot Skill to Workspace

Right-click on the title bar of any robot skill and select a workspace to move to. The current workspace will not be displayed as an option.

Organize Robot Skills with Virtual Workspaces

As a robot project grows, it will have several robot skills that clutter the workspace.
Synthiam ARC can customize virtual workspaces to organize robot skills.
The workspaces are located in ARC's top menu under the File tab. Pressing the ADD or REMOVE buttons lets you add or remove workspaces. Right-click a workspace to rename it.

In this screenshot, we have organized the robot skills by their function.
The Input workspace hosts interactive robot skills, such as joysticks and speech recognition.
The Camera workspace hosts the robot's camera and various tracking and computer vision robot skills.
The Processing workspace hosts scripts and AI Chatbot robot skills.
Finally, the Movement workspace hosts the robot skills that move the robot and interact with the real world.

Right-click Desktop

By right-clicking the desktop, a menu will appear with shortcuts typically found in the top ribbon menu. This pop-up menu lets you add robot skills, view script monitors, stop servos, change workspaces, change project properties, and more.

Loading, Merging & Saving Projects

Auto Load Project

The ARC.exe accepts a few startup parameters.

- Param 1:
Path and Filename of the project to load

- Param 2:
Name of Script to start when project is loaded

***Note:** *Ensure parameters begin and end with quotation marks. This is specifically important when a parameter contains an empty space. Because an empty space is what separates parameters, when the parameter is wrapped in quotation marks, it is parsed as one parameter vs many. Notice in the examples below that the parameters are wrapped in quotation marks.*

Example #1: Loading a project on startup:

"C:\Program Files (x86)\ARC\ARC.exe" "C:\My Documents\MyFile.ezb"

Example #2: loading a project on startup and executing a robot skill script named InitScript

"C:\Program Files (x86)\ARC\ARC.exe" "C:\My Documents\MyFile.ezb" "InitScript"

Shortcut Creator

ARC also includes a shortcut creator so you can have a robot project load when Windows starts, add a shortcut to your desktop, or assign a key combination to quickly load a project. The shortcut creator manual can be found here.

Open Project

There are two ways to open project files (Local file or Cloud Storage). The buttons to access each of these options are in the top File many option tab. Pressing either of these buttons will load their respective dialog windows.

Local File

Loading a local file is for loading files from the hard drive on your PC or network. The Open File dialog will default to the first tab, which is for recent files.
Recent files are the most recent projects that you have opened. The files will continually shift as new files are opened. This is for the convenience of loading your favorite project files.

Other tabs are available, such as "File System", which allows navigating through the file system. The default location of "File System" will be the ARC default project location.
The remaining tabs are categories for various robot types.
Optionally, the auto arrange checkbox can be unselected to prevent robot skills from being auto arranged when the project loads. This remembers the last position all the controls were added.
There is also a checkbox to prevent the Open File menu from loading at startup.

Cloud Files

The Synthiam Cloud provides storage on the internet of your projects, which are also backed up daily. Every time the project is saved to the cloud, a revision history is saved as well. There are filter options on the left of the open file dialog to locate specific files.

Cloud Revision History

Files saved to the cloud with the same filename will maintain revision history.
This allows you to load a project version throughout the lifetime of the project changes.
Locate your file in the cloud dialog and press Version History to view the history.

Auto Arrange Robot Skills

By default, robot skills are auto arranged on workspaces when either a project is loaded, or a new robot skill is added to a workspace. This behavior can be disabled for either scenario by unchecking the option in the respective dialog window.

Loading Robot Project
Adding Robot Skill

Shortcut Creator

ARC includes a shortcut creator so you can have a robot project load when Windows starts, add a shortcut to your desktop, or assign a key combination to quickly load a project. The shortcut creator manual can be found here.

Save Project

Saving files can be done either locally to the computer hard drive or on the Synthiam Cloud for backup and revision history. Saving projects to the cloud allows projects to have a revision history that can be restored.

Save Local

Saving locally means saving the file to the hard drive on the computer.
The default save location is in the My Documents\ARC\My Projects folder.

Save to Cloud

Saving to the cloud will open a dialog for information about the project file.
If this is the first time saving to the cloud, the details will be empty.
Complete the form and submit the file to save to the cloud.
Revisions of projects are associated with the filename.
Continually save the same project using the same filename to use the Synthiam Cloud revision system when opening from the cloud.

Merge Project

Merging projects allows loading an existing project into a new or current project. This option will prompt for selecting what robot skill controls to include in the new project. By using this feature, you may...

- Recover damaged projects
- Extend the capability of existing projects
- Duplicate robot skills in a project

Instructions

Step 1) Locate and click the merge option in the File menu.

Step 2) Navigate and select the project that you wish to import.

Step 3) The first prompt may ask if you wish to import the 3D model. This includes any 3D design files from the import project will replace 3D design files of the current project, including the build instructions.

Step 4) All robot skill controls of the import project will be displayed in a list. Place a check next to each item that you wish to import into the project and press OK.

Completed! The projects have been merged!

Example Projects

ARC contains example projects which demonstrate the functions and features of the software. The example projects can be accessed by using the File Open button on the main

menu.

Press Open

Locate the OPEN button within the FILE section of the top menu. Press it and the Open Dialog window will be displayed.

Open Dialog

The Example Projects are located in the Examples files section. They can be accessed by pressing the Examples button. You may view the contents of each example folder by double-clicking on the folder name within the list. Also, to load a project press the OPEN button. *Note: Pressing OPEN will load a new project, so be sure your current project is saved prior to opening a new one.

Script Examples: This is a folder with examples on how to write code to perform some things. As well as demonstrating some syntax, this folder is more "functional" examples than syntax examples. This also contains examples of every EZ-Script function. For example, if you search for the word "*repeat*" in this folder, you will find an example that shows how to use ez-script *repeat* command.

Legacy Robots: This folder contains older robot projects created by DJ Sures for these robots. Many of these files are outdated and may not load, however, they are included for historical records.

Model Templates: The projects within this folder contain 3D assemblies of the revolution robots. These projects do not include movement panels or any code.

Search

The search field will allow you to search through the files in the current folder.