Robot-Code Draft III

Benjamin Nitkin – January 25th

Overview

R-code standardizes inter-device communication on the robot. R-Code behaves on a master-and-slave framework. The master transmits short codes to either request information or apply settings, and the slave responds with data. The architecture should be more responsive to bandwidth changes than allowing the unit with data to transmit. (i.e. the base station can intelligently prioritize velocity and telemetry over images)

R-Code is roughly based on the G-code of the Reprap and other CAM tools. Two R-code links exist on the robot. The base station acts as a master to a receiving node on the robot laptop. A separate node onboard the robot commands the Arduino. In each case, the master (brain) sends a request that's executed by the slave (body).

Communication

In general, the master sends an R-code with any applicable arguments. The slave transmits a busy flag followed by any applicable data. A ready flag concludes the transmission and clears the master to transmit additional codes.

R-Codes consist of the letter R followed by two numbers, 00 through 99. Expanding into alpha (a-z) will provide additional codes if needed. Arguments are space-delimited, and reside on the same line as the code. `R33 1.0 1.1` is code 33 with arguments 1.0 and 1.1.

The slave response consists of flags and raw data. Flags provide the master with a handshake and information about the robot's current state. Flag's format allows the master to distinguish data from control frames. Flags are semicolon delimited and always follow the format below:

`<timestamp>;<R-code echo>;<Status>`

`<timestamp>` provides time since startup, to at least 1/10th of a second. It's primarily used to time duration of execution and stamp logs.

`<R-code echo>` is the last submitted R-code verbatim. The echo verifies that the correct code was sent, and provides a listening terminal with useful data. The field may be left blank for the `Ready` flag, but should be populated for the `Busy` flag.

`<Status>` is either `Busy` or `Ready`. `Ready` gives the master the go-ahead to transmit codes. `Busy` alerts the master that a code is currently pending, and that any codes transmitted may be ignored. (Codes received while the slave is busy may be processed at the slave's discretion. If processed, the slave should echo a `Busy` flag that includes the R-code echo for each code processed before returning to the `Ready` state.

Notes

This protocol does not use remote echo. In a normal terminal, characters are echoed to the screen as they're typed. Here, there is no feedback until execution of a command, ergo the `<R-code echo>`

The timestamp is given in seconds since the program began: either the ROS master timestamp or, for the Arduino, `millis()/1000`.

For signal processing, JPEG images always start with 0xFFD8 and end with 0xFFD9. JPEG's are likely to be the heaviest data that moves across the radio link.

The regular expression `^[0-9]\.[0-9]*;.*;(Ready|Busy)$` matches flags and only flags. Changing the final `(Ready|Busy)` to either `Ready` or `Busy` will select with more resolution.

ROS is natively metric, so R-codes will be, too. Ranges are in meters and speeds in m/s.

Sample Session

Notable transmissions are bolded. # indicates a comment.

Master Transmits / Slave Receives	Slave Transmits / Master Receives
	0.00;;Ready
#Basic transmission	
R00	
	1.25;R00;Busy
	1.25;;Ready
#Return data	
R10	
	1.43;R10;Busy
	22.4
	1.44;;Ready
#Invalid code	
R29	
	2.00;R29;Busy
	Invalid code.
	2.01;;Ready
#Code with arguments; space-separated	
R33 **1.0 1.0**	
	2.50;R33 1.0 1.0;Busy
	2.51;;Ready
#Set speed in autonomous mode; lockout	
R06	
	2.80;R33 1.0 1.0;Busy
	2.81;;Ready
R33 1.0 1.0	
	3.65;R33 1.0 1.0;Busy
	Autonomous; ignoring.
	3.69;;Ready

R-Code Index

The next page is a complete reference to R-code. The codes are organized by function. R0* provide administrative functions, R2* relates to pathfinding, R5* is used for the IMU sensor suite, and so on.

Left-to-right, the columns are:

1. The code itself, as transmitted by the master
2. A short name for the code
3. Details about what the code does, where helpful
4. The arguments and *(return data)* of each code. See above for implementation of arguments and return values.
5. A sample of the data sent or received, depending on the code. \n is a newline.

R##	Name	Details	Arg 1 ***(Return Data)***	Arg 2	Sample data
R00	Keep-alive	Reset robot watchdog or ping to check if robot alive			
R01	Kill	Software disable motors			
R02	Power down	Power off laptop remotely			
R03					
R04					
R05	Activate teleop	R05-R07 switch active control stations.			
R06	Activate autonomous	Only one of the three control methods can be active at once.			
R07	Activate remote station	Most R-codes work in any mode.			
R08	Check Mode		***Tele, Auto, or Remote***		
R09					
R10	Battery voltage	Retrieve battery voltage from robot	***(Volts)***		22.4
R11					
R12					
R13	Current draw	Measure onboard current, instantaneously	***(Amps, decimal)***		12.4
R14	Average current draw	Retrieve 1-minute average current draw	***(Amps, decimal)***		0.5
R15					
R16	Time since boot	ROS timestamp, measured in seconds	***(Seconds)***		354.78
R17	Status	Describe current status	***(Text status, at author's discretion.)***		Alive.\nTeleoperated.
R18					
R19					
R20	Add waypoint	Append waypoint to robot's navigation list	Latitude	Longitude	
R21	Reset waypoints	Clear navigation goal list			
R22					
R23					
R24					
R25	Retrieve waypoints	Fetch navigation list from robot	***(List of waypoints)***		30.45N\n50.34W\n31.50N\n55.55W
R26	Current waypoint	Fetch current waypoint only	***(Single waypoint)***		30.45N\n50.34W
R27	Next waypoint	Fetch next waypoint	***(Single waypoint)***		30.45N\n50.35W
R28	Planned route	Retrieve the route plotted by the Nav stack	***(10m or so of navigation path)***		Unknown.
R29					
R30	Set twist	Drive the robot by setting motor Twist	???	???	???
R31	Set left speed		Wheel speed, m/s		1.2
R32	Set right speed		Wheel speed, m/s		0.9
R33	Set both speeds		Left wheel speed	Right wheel speed	1.2 1.23
R34	Assigned twist	Retrieve target twist	***(Twist)***		???
R35	Assigned left speed	Retrieve the robot's target speed	***(Speed, m/s)***		1.23
R36	Assigned right speed	Retrieve the robot's target speed	***(Speed, m/s)***		2.31
R37	Actual twist	Retrieve actual twist	***(Twist)***		???
R38	Actual left speed	Retrieve actual wheel speed	***(Speed, m/s)***		1.19
R39	Actual right speed	Retrieve actual wheel speed	***(Speed, m/s)***		2.41
R40	Position	Retrieve GPS position	***(Coordinates)***		30.45N\n50.34W
R41	Accuracy	GPS accuracy, as reported by the Arduino	***(Accuracy, in feet)***		6.2
R42	Number of satellites		***(Number of tracked satellites)***		9
R43	Individual satellite strength		***(List of satellites and their strength)***		2 1.3\n4 2.0 …
R44					
R45					
R46					
R47					
R48					
R49					
R50	Gyro rates	Measure all axes	***(X, Y, and Z slew rates)***		1.23\n2.34\n3.45
R51	Acceleration	Measure all axes	***(X, Y, and Z acceleration rates)***		1.23\n2.34\n3.46
R52	Magnometer	Measure all axes	***(X, Y, and Z magnet readings)***		1.23\n2.34\n3.47
R53	Heading		***(Current heading)***		243
R54					
R55					
R56					
R57					
R58					
R59					
R60	Left range	Retrieve range sensor's reading	***(Range, m)***		20
R61	Center range	Retrieve range sensor's reading	***(Range, m)***		200
R62	Right range	Retrieve range sensor's reading	***(Range, m)***		35
R63					
R64					
R65					
R66					
R67					
R68					
R69					
R70	Image size	Set image size	Image width, pixels	Height, pixels	640 480
R71	JPEG Quality	Set image quality	Quality (0-100)		25
R72					
R73					
R74					
R75	Fetch image	Image comes from predetermined camera (Right, I think.)	***(JPEG image)***		<Raw JPEG bitstream>
R76	Fetch depthmap		***(???)***		???
R77					
R78					
R79					
R80					
R81					
R82					
R83					
R84					
R85					
R86					
R87					
R88					
R89					
R90	Push data	The robot can send an R90, then send arbitrary data.			
R91	Watchdog stop	The robot sends a R91 to tell the base station about a watchdog stop.			
R92	E-Stop	R92 is sent upon an E-stop.			
R93					
R94					
R95					
R96					
R97					
R98					