`roverctl` protocol documentation

JPL Pathfinder Project Division

Apr 4, 1995

1 Introduction

1.1 Purpose

The purpose of the `roverctl` protocol is to control rovers from a long distance. It supports a number of rover features.

1.2 Features

Here is a list of functionality supported by the `roverctl` protocol:

- Telemetry, including system status
- Movement
- Camera
  - Control
  - Image acquisition

2 Protocol

2.1 Overview

The protocol is designed to be as efficient as possible. In addition, the protocol supports a message correction system – any compliant `roverctl` client must wait twenty seconds before acting on a message, as a correction may be sent during that time.

Messages are split into general categories, each corresponding to a rover feature. Thus, when messages are sent, they have a category and an action. Messages are terminated with a series of four null bytes.

2.2 Message format

Messages are sent as raw bytes, without any encoding. Messages use the following format (in hexdump format):

```
ff ff ff ff         # Message preamble
03                  # Category indication (Camera action)
01                  # Message type indication (Rotate camera)
f0 10               # Data (16 degrees counterclockwise)
00 00 00 00         # Message end
```

The message preamble will always be `ff ff ff ff`, and the message end will always be `00 00 00 00`.

2.3 Telemetry

The category byte for Telemetry is `01`. There are two telemetry commands that can be issued.

2.3.1 System status check

The message type indication byte for System status check is `01`. No data may be sent with the request. The client will respond with a System status response message.

2.3.2 Name check

The message type indication byte for Name check is `02`. No data may be sent with the request. The client will respond with with an I am named response message.

2.4 Movement

The category byte for Movement is `02`. There are three movement commands that can be issued.

2.4.1 Halt all movement

The message type indication byte for Halt all movement is `01`. No data may be sent with the request. The client will not send a response.

2.4.2 Set movement speed

The message type indication byte for Set movement speed is `02`. One byte of data should be sent, representing the speed to set, in big-endian. The maximum speed is 255. Setting the movement speed to 0 is equivalent to sending the Halt all movement command. No response will be sent.

2.4.3 Rotate chassis

The message type indication byte for Rotate chassis is `03`. Two bytes of data should be sent. The first four bits sent should indicate the direction to rotate the chassis in. Sending `1111` (hexadecimal `f`) will result in the client turning counterclockwise. Sending `0000` (hexadecimal `0`) will result in the client turning clockwise. The remaining twelve bits should be used to send the amount to turn, in degrees. Sending a value greater than 360 results in undefined behavior. The amount should be sent in big-endian. No response will be sent.

2.5 Camera

The category byte for Camera is `03`. There are two camera commands that can be issued.

2.5.1 Rotate camera

The message type indication byte for Rotate camera is `01`. Two bytes of data should be sent, which represent how much the camera should turn. The amount is specified in the same way as Rotate chassis (see the Movement section). No response will be sent.

2.5.2 Acquire image

The message type indication byte for Acquire image is `02`. No data should be sent. An Image acquisition response will be sent.

2.6 Responses

The category byte for responses is `04`.

2.6.1 System status response

Data for the System status response will be formatted as such. All amounts are in big-endian.

```
ff          # Battery level
10          # Movement speed
```

2.6.2 I am named response

Data for the I am named response will be formatted as such. The string returned is null-terminated and ASCII-encoded.

```
50          # P
41          # A
54          # T
48          # H
```

```
46          # F
49          # I
4e          # N
44          # D
45          # E
52          # R
00          # Null terminator byte
```

2.6.3 Image acquisition

Image acquisition responses will return a grayscale 800x600 image. Each pixel will be represented by a single byte. `00` will represent a black pixel, and `ff` will represent a white pixel. Other values are shades of gray. The image will be sent in rows, then by columns. If coordinates are represented as (row, column), then (0, 0) will be sent first, followed by (0, 1), up to (0, 599), at which the sending will begin again at (1, 0).