RaspWEBCamera 資料

<table>
<tr><td>ポート名</td><td>WebCameraImageOut</td><td>データ型</td><td>CameraImage</td></tr>
<tr><td>出力内容</td><td colspan="3">WEB カメラから画像データを取得し、OutPort から出力します。
出力される画像データのデフォルトのサイズは 320×240 です。</td></tr>
<tr><td>受け取り方</td><td colspan="3">受け取る値は RTC::CameraImage 型で送られてきます。使用するメンバ変数は以下の 3 つになります。
InPort 変数.height： 画像の縦のピクセル数
InPort 変数.width： 画像の横のピクセル数
InPort 変数.pixels： 画像データが入っている 1 次元配列

●C++
送られてきたデータを以下の手順①～③でコピーすることで利用できます。

① データのコピー先の変数（IplImage 型）を定義＆初期化します。
② 関数 cvCreateImage（）を使用し、送られてきたデータの縦：横のサイズと利用するカラーチャンネル分メモリを確保します。
第 1 引数に画像のサイズ、第 2 引数に画像要素のビット数、
第 3 引数にカラーチャンネル数を指定します。

cvCreateImage（CvSize size, int depth, int channels）

Web カメラからの画像をコピーする際は、
第 2 引数に” IPL_DEPTH_8U”（符号なし 8bit 整数） を指定してください。

③ 関数 memcpy（）を使用し、コピー先の変数へデータをコピーします。
このとき、第 1 引数はコピー先のメモリ、第 2 引数はコピー元のメモリ、
第 3 引数にコピーするメモリ領域（長さ）を指定します。

void *memcpy(void *buf1, const void *buf2, size_t n);

送信する際は OutPort 変数のメモリ領域を確保し、送信したい画像データを OutPort 変数にコピー＆出力することで送信できます。

実際の利用方法は下記の C++用参考プログラムを参考にしてください。</td></tr>
</table>

	●Python Python では以下の手順でデータを受け取ります。 ① InPort から受け取ったデータを変数にコピーします。 データのコピーには numpy.frombuffer()メソッドを使用します。 numpy.frombuffer(InPort 変数.pixels, dtype=numpy.uint8) ・frombuffer()メソッドは引数に指定したデータを 1 次元配列で返します。 ・引数 dtype は返す要素のデータ型を指定します。 ② コピーしたデータ配列を（画像の高さ、画像の幅、カラーチャンネル数）のサイズの 3 次元配列に変換します。 変換には numpy.reshape()メソッドを使用します。 numpy.reshape(画像の高さ、画像の幅、カラーチャンネル数) これらの手順を行うことで、OpenCV を使用した画像加工が利用できます。 送信する際は OutPort 変数に以下の様にデータを渡します。 OutPort 変数.height = InPort 変数.height OutPort 変数.width = InPort 変数.width OutPort 変数.pixels = 送信データ変数.tostring() 実際の利用方法は、下記の Python 用参考プログラムを参考にしてください。
参考プログラム	
C++用：https://rtc-fukushima.jp/wp/wp-content/uploads/2017/07/20170825_3_tejun6.pdf	
Python 用：https://rtc-fukushima.jp/wp/wp-content/uploads/2019/01/FlipComp.zip	