1 ソフトウェア

1.1 画像認識

炎上ポールの認識には配布された Rapspberry Pi NoIR Camera V2（以下，カメラモジュール）を使用する．画像撮影から最も近いポールと思われる物体へのベクトルを出力する一連の手順を，簡単に以下に示す．画像処理には OpenCV ライブラリを用いており，各処理で使用した主要なライブラリ関数を併記する．

画像取得

カメラモジュールへのアクセスには OpenCV とは異なる既成ライブラリ [1] を利用した．撮影によりピクセル値の 2 次元配列（cv::Mat）が出力として得られる．

カラーモデル変換

得られた画像のカラーモデルを RGB から HSV に変更する．

使用関数：`cv::cvtColor`

2 値画像化

HSV 画像データに対し赤色マスクをかけて 2 値画像に変換する．

使用関数：`cv::inRange`

ノイズ除去

モルフォロジー処理によりノイズを除去する．

使用関数：`cv::morphologyEx`

構造解析

2 値画像中の輪郭線を検出した後，それを矩形で囲む．囲んだ矩形を縦横比で解析し，ポールの縦横比に対して ±20% 以上の差があるものを除外する．

使用関数：`cv::findContours`，`cv::boundingRect`

ベクトル作成

除外されず残った矩形の重心点を求め，カメラの画角（62.2×48.8 [2]）を元に機体中心から重心点へ向かうベクトルを作る．

使用関数：`cv::moments`

1.2 進行方向

機体周囲に放射状に取り付けられた測距センサによる距離情報をもとに障害物回避を行う．

まず，スカラーである距離情報に方向の情報を付加するために，実際の機体周囲の測距センサの配置を元に，次のように 8 つの二次元単位ベクトルを定義する．

$$\boldsymbol{u}_0 = \begin{bmatrix} -0.707 \\ -0.707 \end{bmatrix}, \quad \boldsymbol{u}_1 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \quad \boldsymbol{u}_2 = \begin{bmatrix} 0.707 \\ -0.707 \end{bmatrix}, \quad \boldsymbol{u}_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

$$\boldsymbol{u}_4 = \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}, \quad \boldsymbol{u}_5 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \boldsymbol{u}_6 = \begin{bmatrix} -0.707 \\ 0707 \end{bmatrix}, \quad \boldsymbol{u}_7 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

障害物との衝突を避けるため，センサにより得られた距離が小さいほど大きな斥力が機体の進行方向に作用するようにしなければならない．また同時に，離れすぎることも競技エリア外に出てしまうことが懸念されるため避けなければならない．そのため，距離が大きい時には大きな負の斥力，すなわち障害物からの引力を受けるように，次の式を利用して距離情報を『危険度』の情報に変換する．

$$y = -\tanh^{-1}(x-1) \tag{1.1}$$

式（1.1）を図に表すと図 1.1 となる．

実際には適正距離で $\tanh^{-1}(x-1) = 0$ となるように，維持したい距離とセンサの最大レンジないしは採用する最大の距離を考慮に入れて，それらを正規化した値を代入しなければならない．

また，零距離から適正距離，適正距離から最大距離の比が 1 : 1 とならない場合は，場合を分けて正規化を行わなければならない．

以上の処理により近接センサ 3 つと PSD センサ 7 つの距離情報から作成したベクトルを合成し，正規化する．前節の画像処理により画像内に赤色の物体が検知されていた場合はそれに向かうベクトルに，検知されていなかった場合は自己位置推定の結果から順路に沿ってゴールへ向かうベクトルを作成し，それに距離情報から作成したベクトルを合成し，進行方向とする．

最後に，進行方向のベクトルが左右のモータへ与える速度指令値に変換されて入力される．

参考文献

[1] ”RaspiCam: C++ API for using Raspberry camera with/without OpenCV”,
”https://www.uco.es/investiga/grupos/ava/node/40”,
2017 年 5 月 31 日最終確認.

[2] ”RPi Camera Module - eLinux.org”,
”http://elinux.org/Rpi_Camera_Module#Technical_Parameters_.28v.2_board.29”,
2017 年 5 月 31 日最終確認.

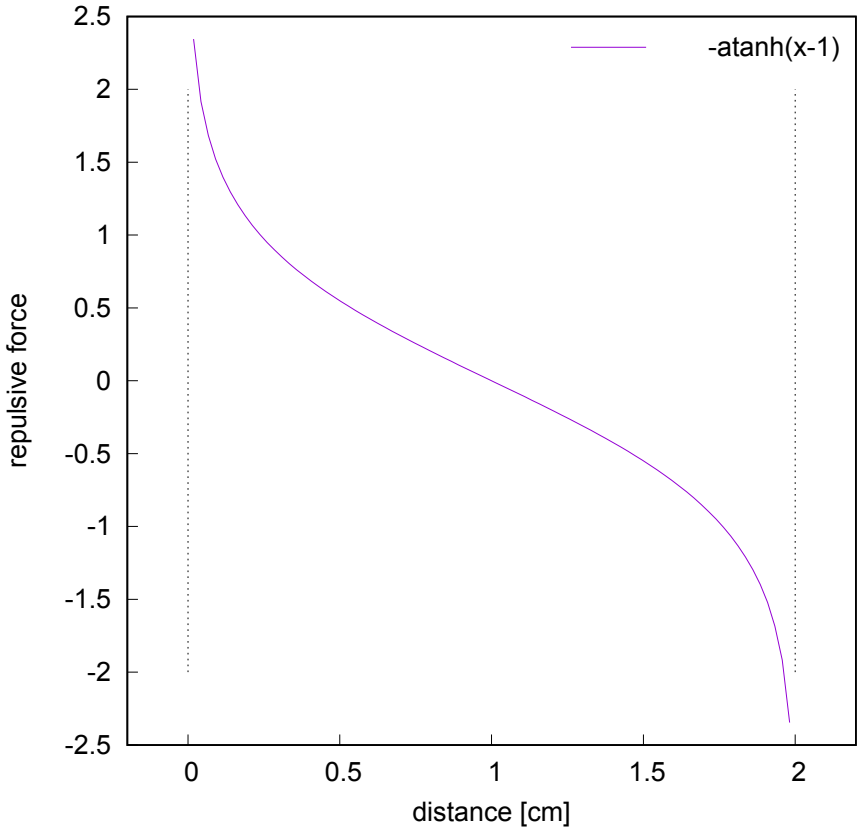

図 1.1　$y = -\tanh^{-1}(x-1)$