開発レポート　Development Report

ラントレースソフト「CWM_CamLine」の開発

― 床のラインと赤マーカーをカメラで認識して走る ―

CWM_CamLine: Camera-Based Line Following and Red-Marker Stop for a Mobile Robot

レポート番号	MTIC-DR-2026-006
団体	NPO 法人松戸技術革新センター M-TIC　松戸ロボット研究会
著者	大城 公平
発行日	2026 年 10 月 25 日
対象	CWM_CamLine v0.1.2（ソフトウェア）
分類	開発レポート／画像認識・走行制御ソフト
機体状態の基準日	2026 年 9 月 21 日

本レポートの数値と機体状態について　本検証系は研究開発段階にあり随時変更されるため、記載の仕様は基準日時点のものであり、現在の実機と一致することを保証しない。数値には種別を付す：**公称**＝カタログ・設計値、**実測**＝実際に計測した値（記載桁が測定分解能）、**概算**＝目測・推定値。

概要

床に引いたラインに沿って走るライントレースは、移動ロボットの基本的な走行方法である。従来は機体の底面に並べた赤外線センサー（IR アレイ）でラインを検出していたが、センサーの真下しか見えないため、先のカーブを予測できない。「CWM_CamLine」は、M-TIC が研究開発しているピッキングロボット CWM（水平多関節アームで物をつかんで運ぶロボット）のインホイール走行を、USB カメラの映像でライントレースさせる専用ソフトである。カメラ画像の下部から床のラインを抽出して左右のずれを求め、赤色のマーカーを検出すると停車する。ライン位置は既存の赤外線センサーと同じ形式で機体制御ソフトへ渡すため、機体制御ソフトを改変せずにセンサーを置き換えられる。赤外線センサーから画像認識に変えることで、ラインテープの幅や色を用途に合わせて変えられるなど、物理的なセンサーより対応できる範囲が広がる可能性がある。一方で、画像処理の遅れや認識の揺れにより、比例制御が赤外線センサーのときほど滑らかに効かず、走行がかえって不安定になる懸念もある。PC 上の試験では、処理解像度 320×240 で毎秒 71〜83 フレームの処理速度を確認し、直線・カーブ・赤マーカー・ラインの終端をそれぞれ判定できた。試験は白い床に黒いラインを貼った条件で行ったもので、CWM に搭載した走行試験はこれからである。

1. 背景と目的

1.1 開発背景

CWM は、M-TIC が研究開発している、水平多関節（SCARA 型）アームで物をつかんで運ぶピッキングロボットで、インホイールモーターで走行する。棚や作業場所の間を決まった経路で移動し、指定の位置で止まって物を扱うことを想定している。ライントレースには底面の赤外線センサーを用いてきたが、赤外線センサーはセンサーの真下のラインしか検出できず、カーブの手前で減速するといった先読みができない。M-TIC ではカメラ認識の検証アプリ（Vision Studio、MTIC-DR-2026-004）でライントレースモードを試作しており、これを CWM の走行専用に切り出した。

また、赤外線センサーは床とラインの反射率の差でラインを検出するため、使えるラインの色や幅はセンサーの特性と並び方で決まる。画像認識であれば、ラインの色や幅の条件をソフトの設定で変えられるため、既存の床に合わせたラインを選べるなど、使える場面が広がると考えられる。

1.2 目的と目標

カメラの映像だけでライントレースと停車位置の判定を行い、既存の赤外線センサーを置き換えられるソフトを作る。機体制御ソフトは改変しないことを条件とする。

目標：CWM がカメラだけでライン 1 周を逸脱せずに走り、赤マーカーで停車できること。

2. システム構成

2.1 開発環境と基本的な作り方

開発は Python と Thonny で行った。ライントレース専用なので、Vision Studio から必要な処理（OpenCV による 2 値化と重心計算、Tkinter の確認画面）だけを切り出し、ライン位置は UDP で機体制御ソフトへ渡す。機体側は赤外線センサーのときと同じ値を受け取るだけで済む（図 1・図 2）。



図 1　開発環境と基本構成



図 2　CWM_CamLine の処理の流れ

※ 本レポートでは処理の考え方と構成のみを示し、ソースコードおよび調整値は掲載しない。

2.2 ラインと赤マーカーの認識の考え方

カメラ画像の下半分に、手前・中間・奥の 3 段の検出帯を設ける。各帯を白黒の 2 値に変換すると、床とラインの明るさの違いからラインだけが取り出せる。各帯でラインの重心（中心位置）を求め、3 つの重心を結んだ線から「ラインの左右のずれ（位置）」と「ラインの傾き（角度）」を得る。ステアリングは、位置のずれ・角度・ずれの変化量を組み合わせて決める（位置と変化量を使う方法は PD 制御と呼ばれる基本的な考え方である）。停車位置の赤マーカーは、色を「色相・彩度・明度」で表す HSV 色空間に変換して赤い範囲だけを取り出し、一定以上の面積があれば停車と判定する。いずれも OpenCV の基本的な機能の組み合わせで実現できる。

認識結果は「CENTER（中央）」「LEFT／RIGHT（左右にずれ・カーブ）」「MARK（赤マーカー）」「END（ラインが見えない＝終端）」などの状態として画面に大きく表示し、認識の安定度（揺れ）も表示と音で知らせる。モーターを動かさずに指令値だけを確認できるシミュレーション機能、照明のむらに強い 2 値化、黒い床の白線にも対応する切り替えを備える（図 3）。

Copyright © 2026 Matsudo Technology Innovation Center. All rights reserved.

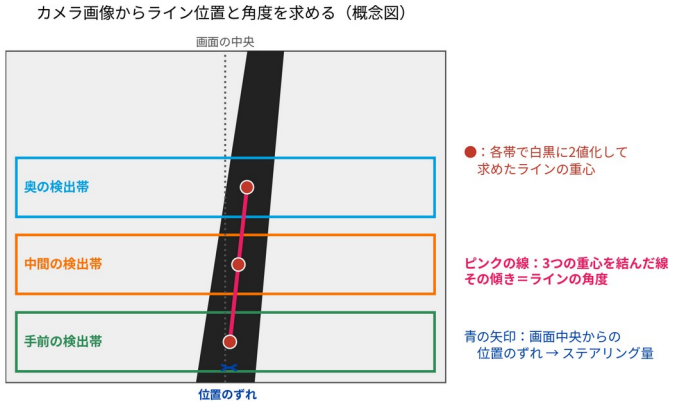


図３　カメラ画像からライン位置と角度を求める（概念図）

項目	内容
版	v0.1.2
搭載先	CWM（Raspberry Pi 4B ＋ USB カメラ）
カメラ	USB ウェブカメラ（Logicool 製、型式 V-U0018、720p、5 V 500 mA、公称）。処理は 320×240 に縮小して行う
入力	USB カメラの床の映像（進行方向前方）
出力	ライン位置（既存の赤外線センサーと同じ形式）、停車判定
接続	機体制御ソフトとは別プロセス。UDP で接続し、機体制御ソフトは改変しない

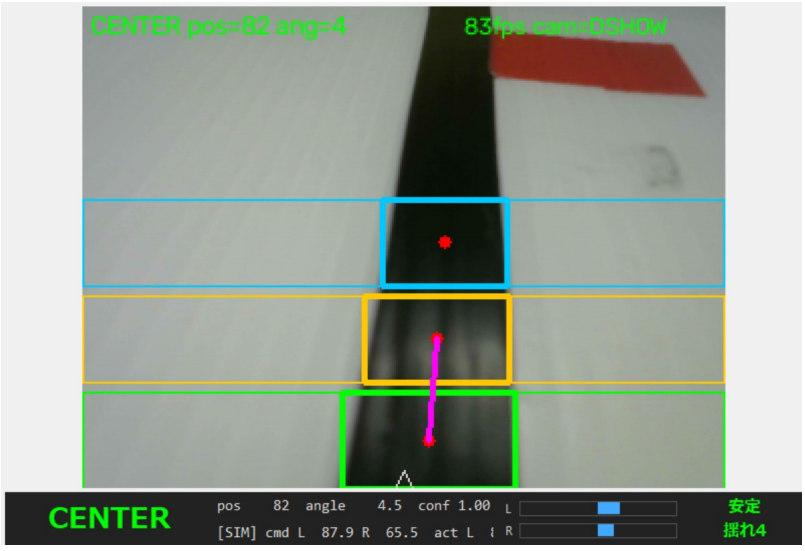


図４　ライン追従中の画面（CENTER）。３段の検出帯と重心を結んだ線、状態・安定度を表示

Copyright © 2026 Matsudo Technology Innovation Center. All rights reserved.

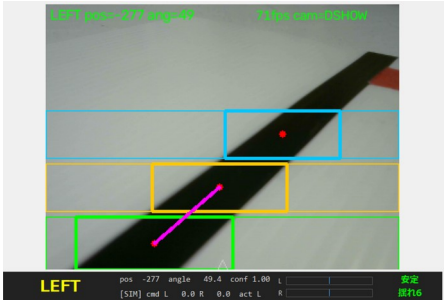


図 5　左カーブの判定（LEFT）

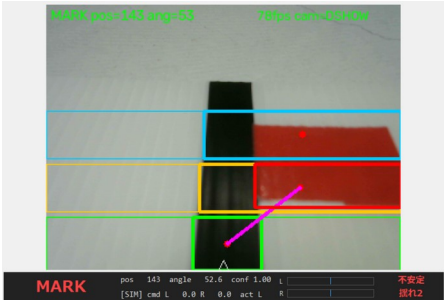


図 6　赤マーカーの検出（MARK）

※ 画面は PC 上で取得。調整値の表示部分は掲載していない。

2.3 赤外線センサーとの比較

項目	赤外線センサー（従来）	カメラ（本ソフト）
見える範囲	センサーの真下のみ	機体の前方まで（先読みが可能）
停車マーカー	別のカラーセンサーが必要	同じカメラで判定できる
光の影響	受けにくい	照明・影・反射の影響を受ける
処理	軽い	画像処理のため計算量が多い
ラインの色・幅	反射率の差とセンサーの並び方で決まり、変えにくい	設定で変えられる（黒線・白線の切り替え、幅の異なるテープなど）
制御の滑らかさ	センサーの出力をすぐに比例制御に使える	画像処理の遅れと認識の揺れがあり、比例制御が滑らかに効かない懸念がある

3. 検証方法と結果

3.1 検証方法

Windows PC に USB カメラを接続し、白い床に貼った黒いラインと赤いマーカーを手で持ったカメラで映して、各状態の判定と画面表示、処理速度を確認した。モーターは動かさず、シミュレーション機能で左右の指令値の出力を確認した。

3.2 結果

項目	結果
処理速度（PC、320×240）	毎秒 71〜83 フレーム（実測、画面表示）
直線・左右のずれ	CENTER／LEFT／RIGHT を判定（図 4・図 5）
カーブ	ラインの傾き約 49°を角度として検出し LEFT と判定（図 5）
赤マーカー	MARK を判定（図 6）
ラインの終端	ラインが映らないと END を判定し、指令値を 0 にする
指令値の出力（シミュレーション）	追従中に左右の指令値が出力されることを確認

CWM に載せた実走行と、Raspberry Pi 4B での処理速度は、機体の完成を待って測る（5.1 節）。

Copyright © 2026 Matsudo Technology Innovation Center. All rights reserved.

4. 課題と考察

- **照明の影響：**2 値化は床とラインの明るさの差に頼るため、影や照明の映り込みでラインが途切れて見えることがある。環境ごとに 2 値化の基準を調整する必要がある。
- **処理の遅れ：**PC では毎秒 70 フレーム以上で処理できたが、Raspberry Pi 4B では処理能力が低いため速度が下がると見込まれる。処理が遅れるほどステアリングの反応が遅れ、走行速度を上げるほど影響が大きい。
- **比例制御への影響：**カメラ方式でもラインの位置は数値として得られるため、比例制御そのものは行える。ただし、位置の値はフレームごと（PC で毎秒 70 回程度）にしか更新されず、画像処理の遅れや認識の揺れも加わるため、赤外線センサーで行っていたときほど滑らかに制御が効かず、蛇行するなど走行がかえって不安定になる懸念がある。
- **カメラの揺れ：**カメラを手で持った試験でも認識が不安定になる瞬間があった（図 6 の「不安定」表示）。走行中の振動でも同様の揺れが起こりうるため、カメラの固定方法の検討が必要である。
- **実機での確認：**CWM に搭載した走行試験を行っておらず、赤外線センサーと同等以上に走れるかは未確認である。

5. 今後の展望

5.1 検証計画

計測項目	方法	判定基準
追従精度	CWM でコースを 10 周走行し、逸脱回数を記録	逸脱ゼロ
停車成功率	赤マーカーでの停車を 10 回試行	成功回数／10
照明への強さ	昼・夜・窓際で同じ試験	各条件で逸脱ゼロ
赤外線センサーとの比較	同じコースで両方式の周回時間と蛇行の大きさを比較	カメラ方式の周回時間が同等以下、蛇行が同等以下
ラインの色・幅の違い	色（黒・白・色付き）と幅の異なるテープで同じ試験	各条件で逸脱ゼロ

5.2 応用の可能性

ラインの色や幅を設定で変えられる特長を生かせば、色の違うラインで経路を分けたり、既存の床の模様に合わせたラインを使ったりすることも考えられる。カメラの先読みを使えば、カーブの手前での減速や分岐の判別もできると考えられる。ラインのない場所でも床面を認識して走る M-TIC AutoDrive（MTIC-DR-2026-003）と組み合わせ、ラインのある区間とない区間を切り替えて走る構成へ発展させる。

5.3 カメラ認識技術の統合（構想）

M-TIC ではカメラを使った認識技術を、床面認識（M-TIC AutoDrive）、ライン認識（CWM_CamLine）、物体検知（M-TIC Detect）、色・形の認識（Vision Studio）として、それぞれ単体で開発・検証してきた。今後はこれらを最終的に組み合わせることで、自動運転の技術に使える可能性がある。たとえば、床面認識で走れる範囲を把握しながら物体検知で走行中の障害物を見つけて避ける、人がロボットの経路を横切ったときに物体検知で人と判定して減速・停止する、ラインのある区間ではライン認識、ない区間では床面認識に切り替えて走る、といった使い方が考えられる。まずは単体ごとの精度と処理速度を確かめたうえで、組み合わせ方を段階的に検討していきたい。組み合わせの例を図 7 に示す。

Copyright © 2026 Matsudo Technology Innovation Center. All rights reserved.

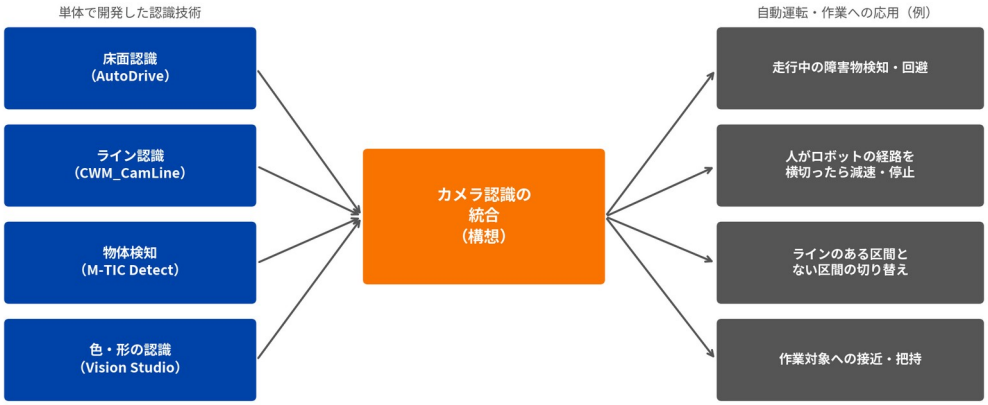


図 7　カメラ認識技術の統合の構想（各レポートの技術を組み合わせた応用の例）

6. まとめ

カメラの映像から床のラインと赤マーカーを認識し、既存の赤外線センサーと同じ形式でライン位置を出力するライントレース専用ソフト CWM_CamLine v0.1.2 を開発した。PC 上の試験で、毎秒 71〜83 フレームの処理速度と、直線・カーブ・赤マーカー・終端の判定を確認した。機体制御ソフトを改変せずにセンサーを置き換えられる構成とした。CWM での走行試験が次段階の課題である。

参考資料

OpenCV ドキュメント（opencv.org）：Thresholding, Color Spaces, Image Moments

改訂履歴　初版（2026-09-30）　2026-10-25　文章を見直し、誤記を訂正。

Copyright © 2026 Matsudo Technology Innovation Center. All rights reserved.