

MATLABではじめる自律移動システム開発 ～SLAMとパスプランニング～

MathWorks Japan
アプリケーションエンジニアリング部
木川田 亘

搬送車の自律移動制御(SLAMによる地図生成とパスプランニング)

The image is a composite of three parts:

- Top Left:** A screenshot of the MATLAB R2018a interface. The main window displays a presentation slide titled "パスプランニングによる自動駐車デモ" (Automated Parking Demonstration by Path Planning). The slide includes a Simulink block diagram, a graph of path occupancy, and a 3D Gazebo simulation of a yellow mobile robot. Below the slide, there is Japanese text: "パスプランニングアルゴリズム開発におけるMATLAB/Simulink®活用方法をご紹介します。" (We introduce the usage of MATLAB/Simulink® in the development of path planning algorithms.) and "例として自動駐車の場合、" (For example, in the case of automated parking, "作成" (creation) "による経路生成" (path generation) "ローラによる経路追従。" (path following by rollers).
- Bottom Left:** A 3D rendering of a yellow mobile robot with a camera and a sensor mounted on top, positioned in front of the MATLAB window.
- Right:** A virtual machine desktop environment. A terminal window displays the IP address "192.168.112.130" and login instructions: "For administrative changes", "User name: user", and "Password: password". The desktop background is blue and features several application icons: "Gazebo Playground", "Gazebo PR2 Simulator", "Gazebo LBR Simulator", "Gazebo Husky Playpen", and "Gazebo Jackal World". There is also a "Terminal" icon and a "Kubuntu_OS_rights" folder icon.

事例 : Clearpath Roboticsが産業用ロボットのアルゴリズム開発を加速

課題

産業用ロボットのレーザーベースの認識、
コンピュータービジョン、走行管理、
制御アルゴリズムを短期間で開発する

解決策

MATLABを、ROSデータの可視化と解析、プロトタイピング、
最新のロボット研究のアルゴリズムの適用に活用

結果

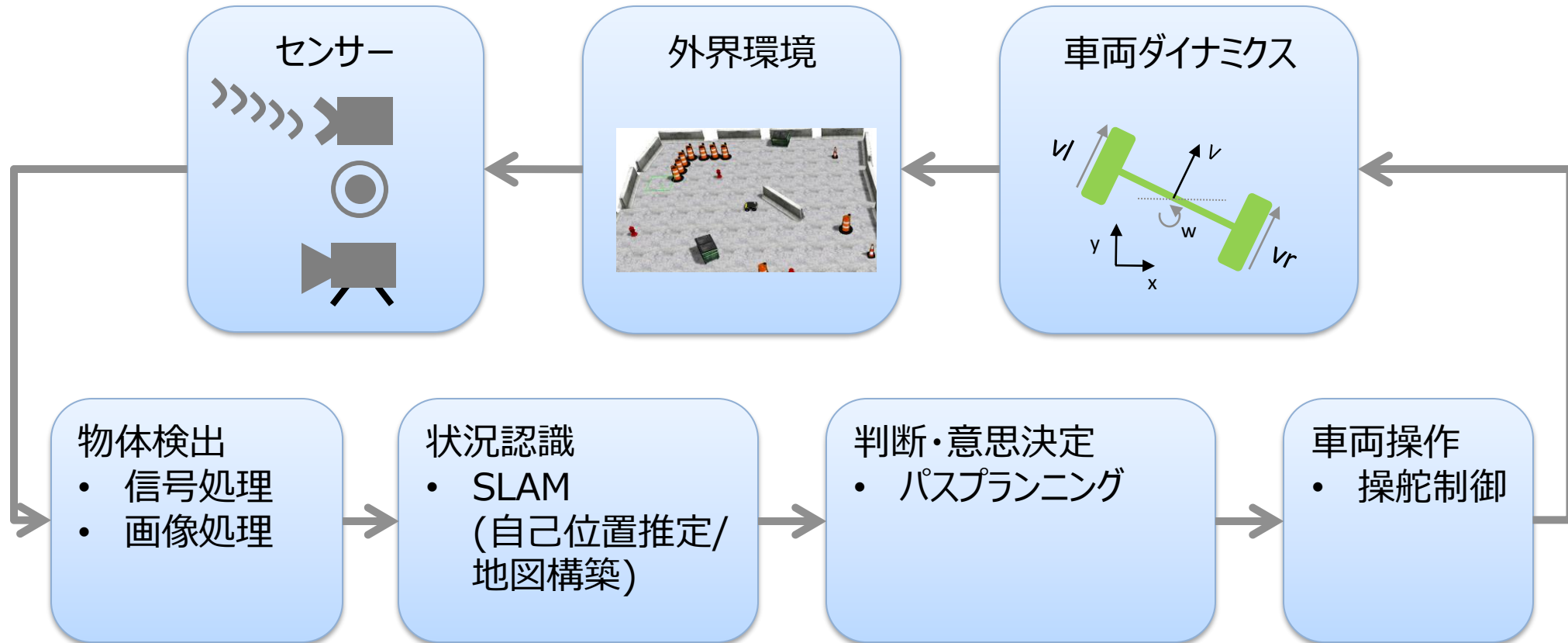
- データ解析時間が最大50%短縮
- 顧客とのコミュニケーションを効率化
- 最新の自動運転アルゴリズムを素早く組み込み



An OTTO self-driving vehicle from Clearpath Robotics.

“ROS is good for robotics research and development, but not for data analysis. MATLAB, on the other hand, is not only a data analysis tool, it’s a data visualization and hardware interface tool as well, so it’s an excellent complement to ROS in many ways.”
- Iliia Baranov, Clearpath Robotics

自律移動システムの開発検証



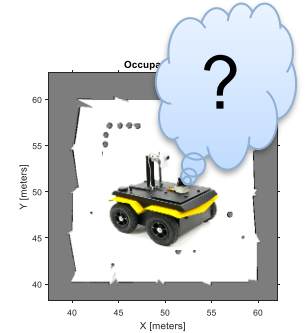
自律システムはClosed-loopで構成される複雑なシステム
システムレベルでの妥当性の検証が必要

パスプランニングアルゴリズムの開発フロー

SLAM(自己位置推定/地図構築)
(アルゴリズム開発に必要な周辺要素)



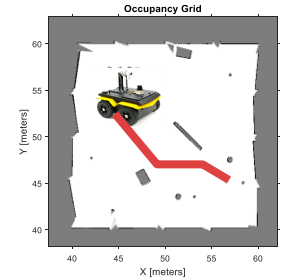
障害物表現
センサー信号処理
状態推定



パスプランニング
(アルゴリズム単体開発・検証)



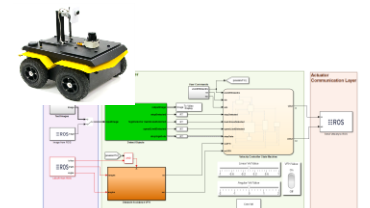
経路計画アルゴリズム
最適化計算



システムへの統合
(システムレベルの検証)



コンポーネントの制御ロジック
環境シミュレーション
自動コード生成&配布



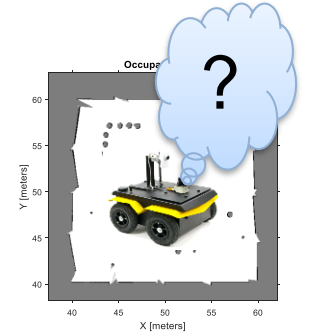
MATLAB/Simulinkを活用することで単一環境で自律システム開発

パスプランニングアルゴリズムの開発フロー

SLAM(自己位置推定/地図構築)
(アルゴリズム開発に必要な周辺要素)



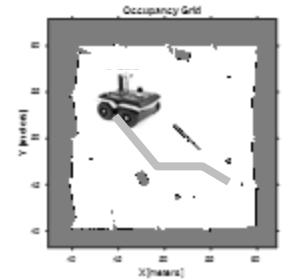
障害物表現
センサー信号処理
状態推定



パスプランニング
(アルゴリズム単体開発・検証)



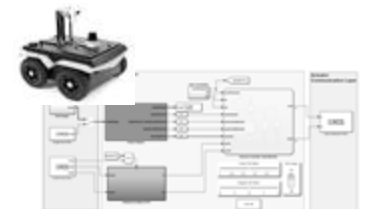
経路計画アルゴリズム
最適化計算



システムへの統合
(システムレベルの検証)

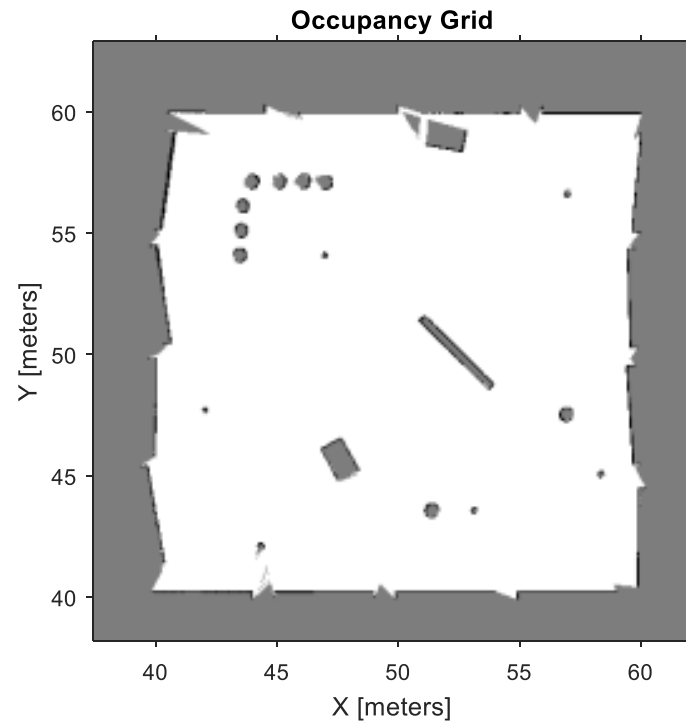


コンポーネントの制御ロジック
環境シミュレーション
自動コード生成&配布



さまざまな地図表現

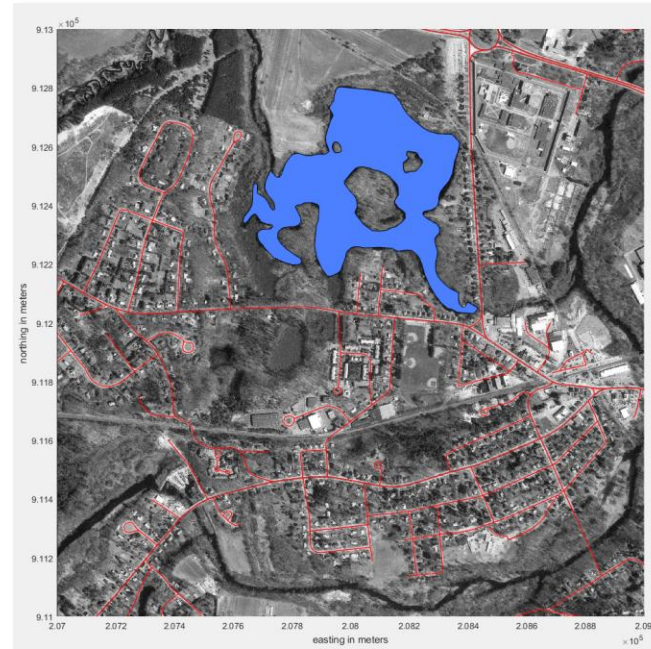
2Dマップ
(グリッド/ベクター)



障害物を0~1で表現
プログラムでの取り扱いが容易

Robotics System Toolbox

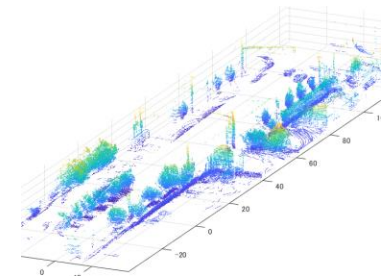
2Dマップ
(航空写真/ベクター)



一般的な地理データ
道路や建物なども表現

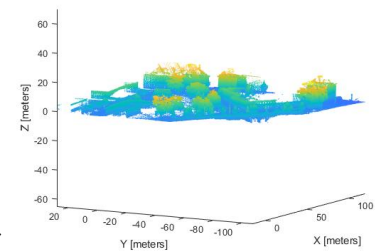
Mapping Toolbox™

3Dマップ
(グリッド/点群/ベクター)



点群

Computer Vision System Toolbox™



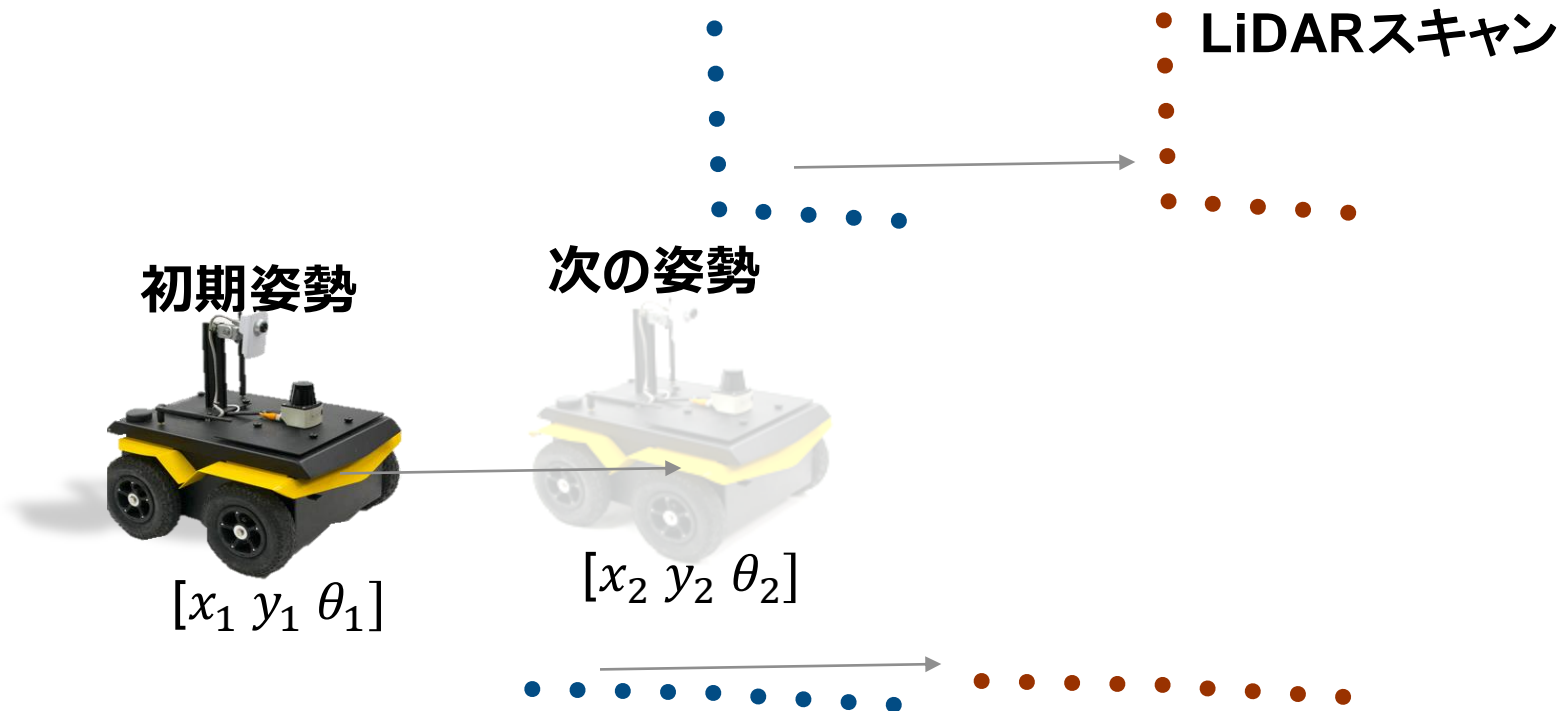
グリッド

Robotics System Toolbox™ R2018a

点群やベクターデータ
より高精細な地図

SLAMによる地図構築

- SLAM(Simultaneously Localization and Mapping)とは？
 - 自己位置推定と地図作成を同時に行う方法
 - LiDAR(レーザーセンサー)や複数のセンサーを統合して推定



使用するセンサー



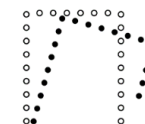
ホイールエンコーダー
(Odometry)



慣性計測装置
(IMU)



GPS



スキャンマッチング

LiDARの点群データの位置合わせ

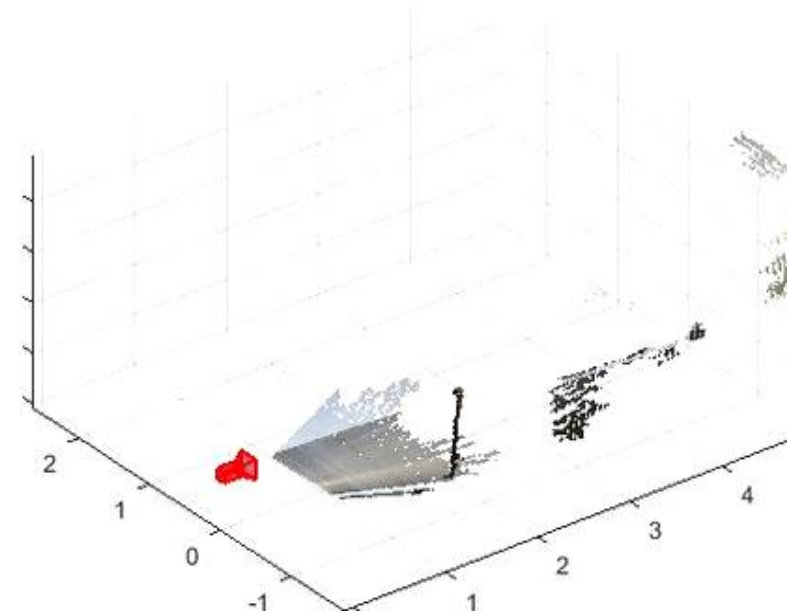
Robotics System Toolbox™

- 2Dのスキャンマッチング
 - [matchScans](#) (R2017a)
 - [matchScansGrid](#) (R2018a)



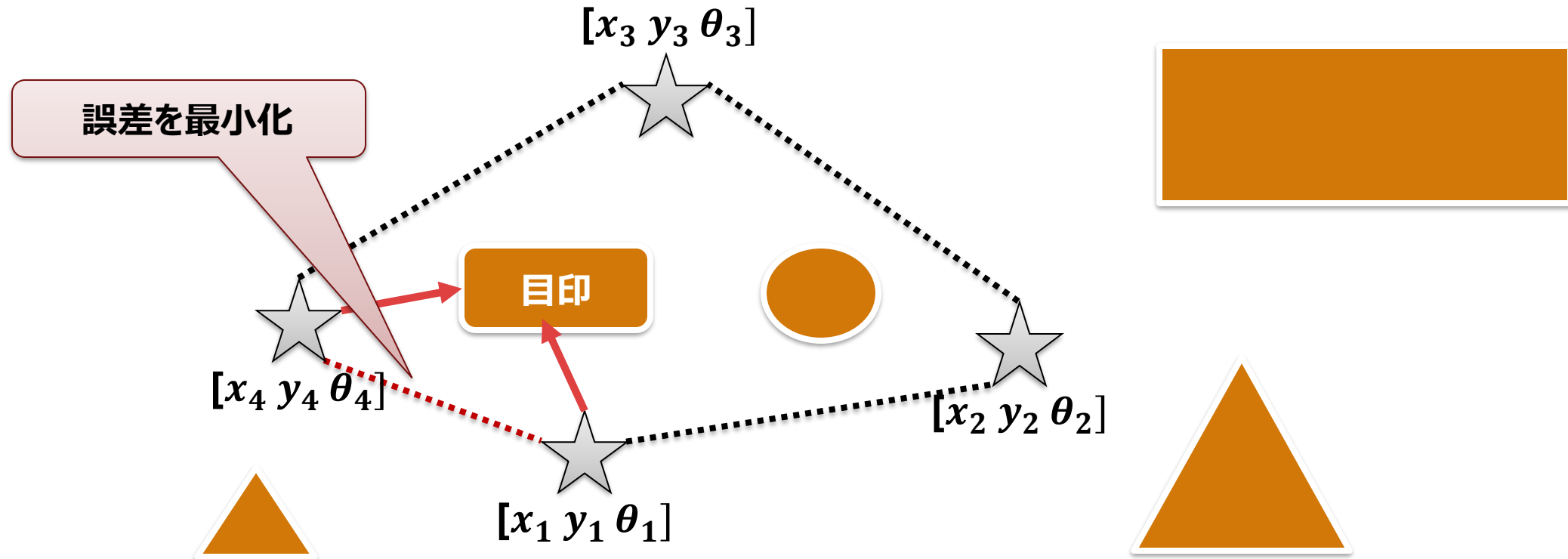
Computer Vision System Toolbox™

- 3Dのスキャンマッチング
 - [pcregistericp](#)
(R2018aでpcregrigidから名称変更)
 - [pcregisterndt](#) (R2018a)

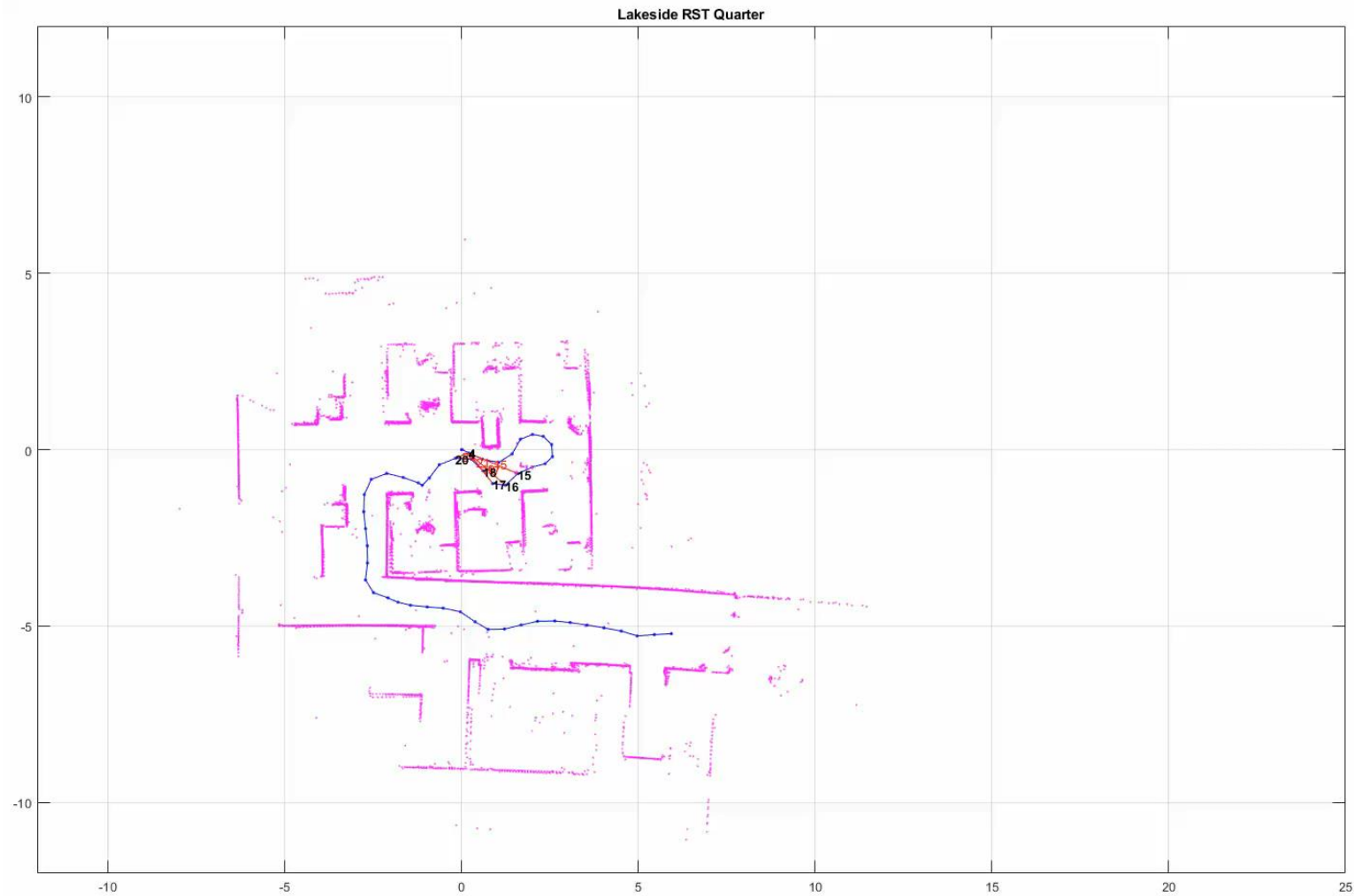


SLAMの閉ループ問題

- 誤差が蓄積すると自己位置が正しく推定できない
- ロボットが経由した場所を目印として保持⇒ポーズグラフ(姿勢グラフ)
- 最適化で誤差の蓄積(ドリフト)を抑える



LiDAR SLAMによるループの閉じ込み

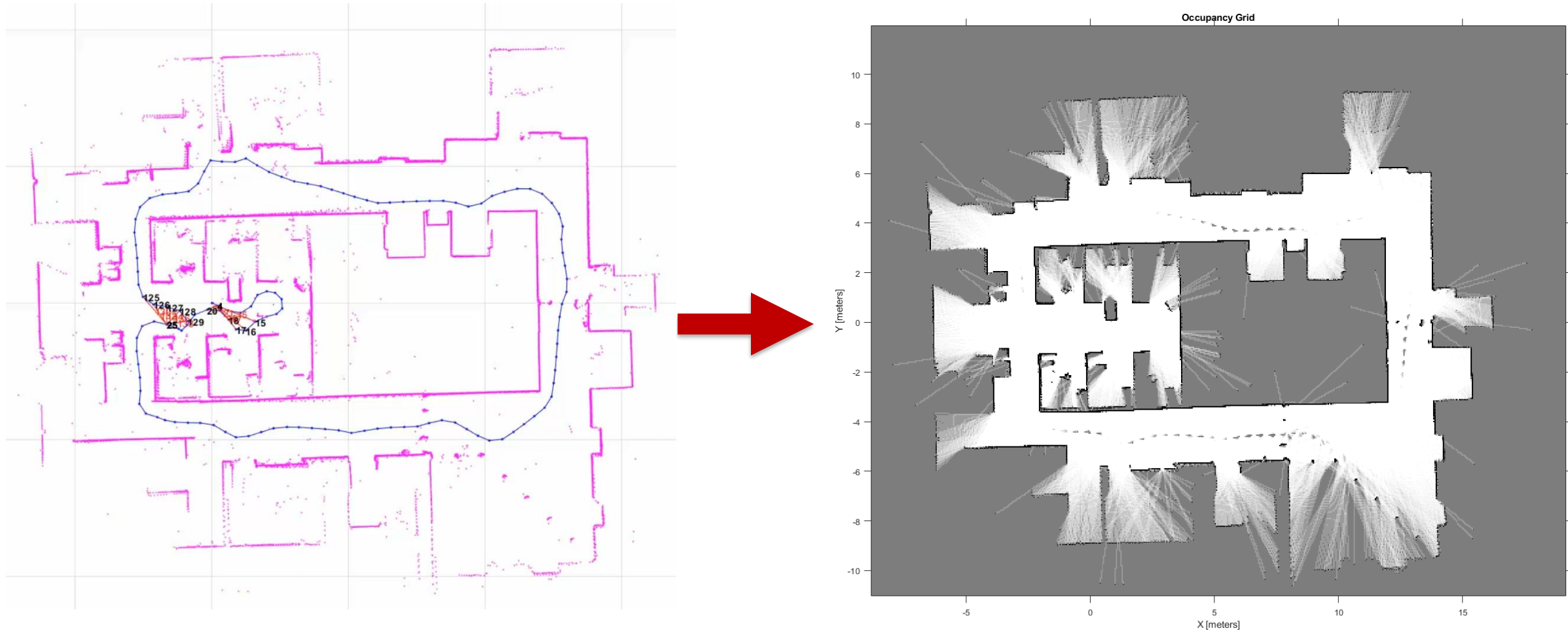


```
robotics.LidarSLAM
```

R2018a

Robotics System Toolbox™

地図データの生成：スキャンデータをグリッドマップ化

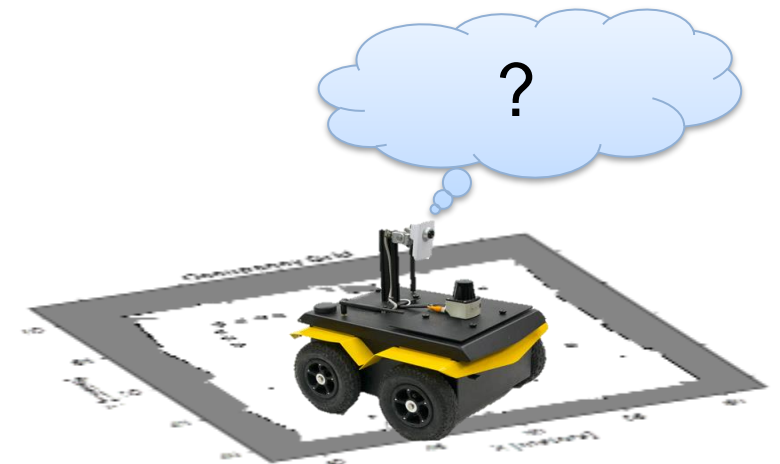
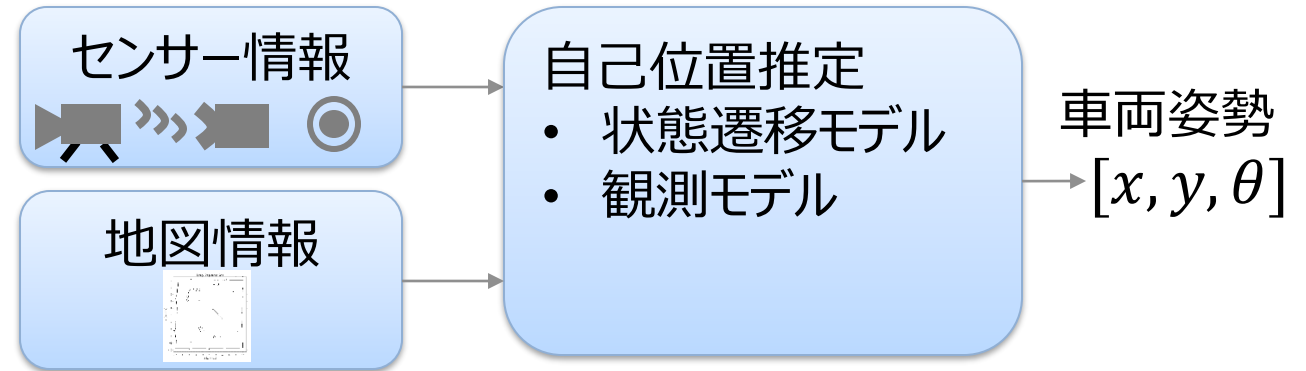


buildMap

R2018a Robotics System Toolbox™

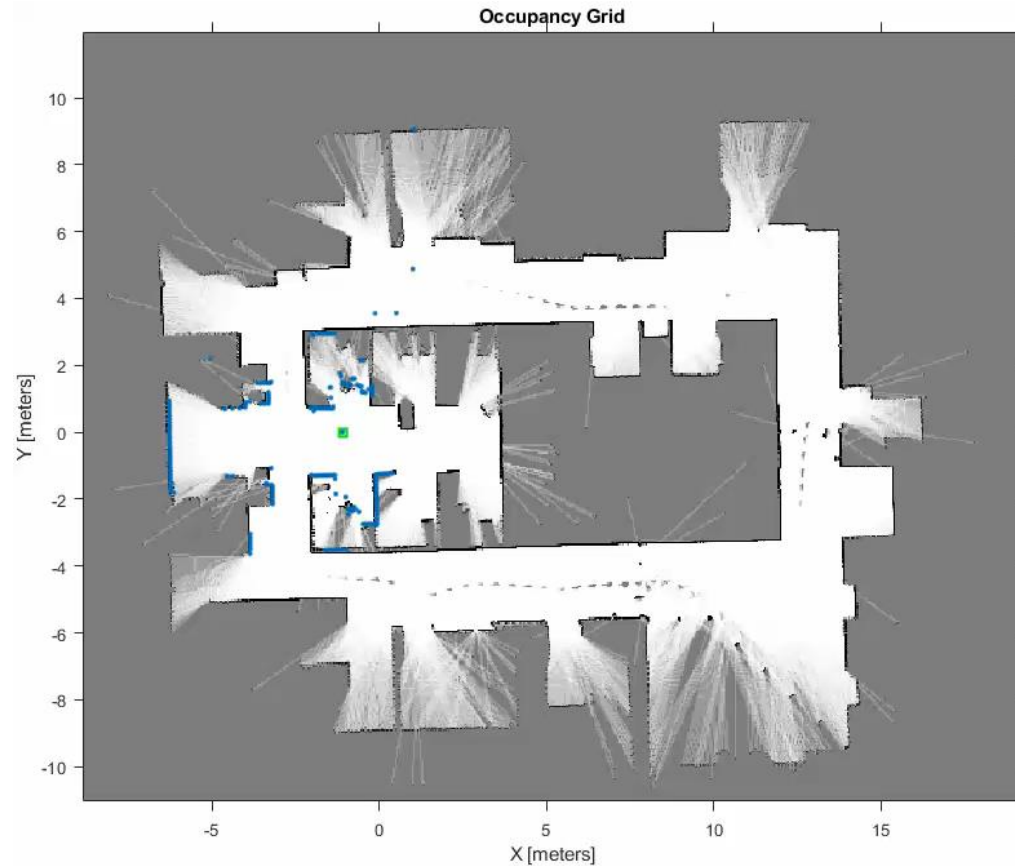
自己位置推定とは？

- 自己位置推定(Localization)とは
地図上のロボットの姿勢を
センサー情報を基に推定する技術
- LiDARによる姿勢推定
 - スキャンマッチング
 - ICPアルゴリズム*1
 - **NDTアルゴリズム*2**
 - ベイズフィルタ
 - **パーティクルフィルタ*2**
(モンテカルロローカリゼーション)
 - カルマンフィルタ
(線形/拡張/Unscented) *3



自己位置推定のアルゴリズム

Robotics System Toolbox™

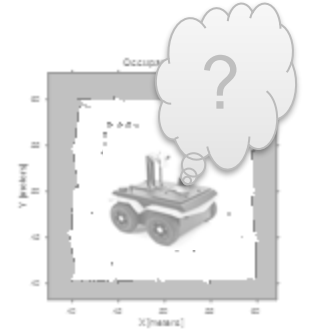
`matchScansGrid`**R2018a**`robotics.MonteCarloLocalization`

パスプランニングアルゴリズムの開発フロー

SLAM(自己位置推定/地図構築)
(アルゴリズム開発に必要な周辺要素)



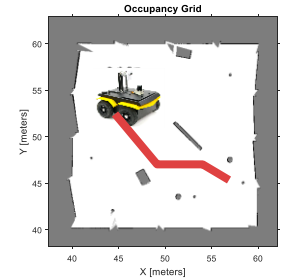
障害物表現
センサー信号処理
状態推定



パスプランニング
(アルゴリズム単体開発・検証)



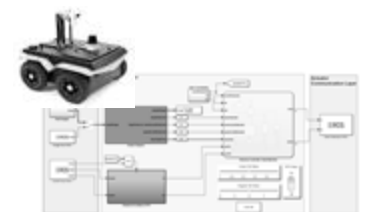
経路計画アルゴリズム
最適化計算



システムへの統合
(システムレベルの検証)



コンポーネントの制御ロジック
環境シミュレーション
自動コード生成&配布

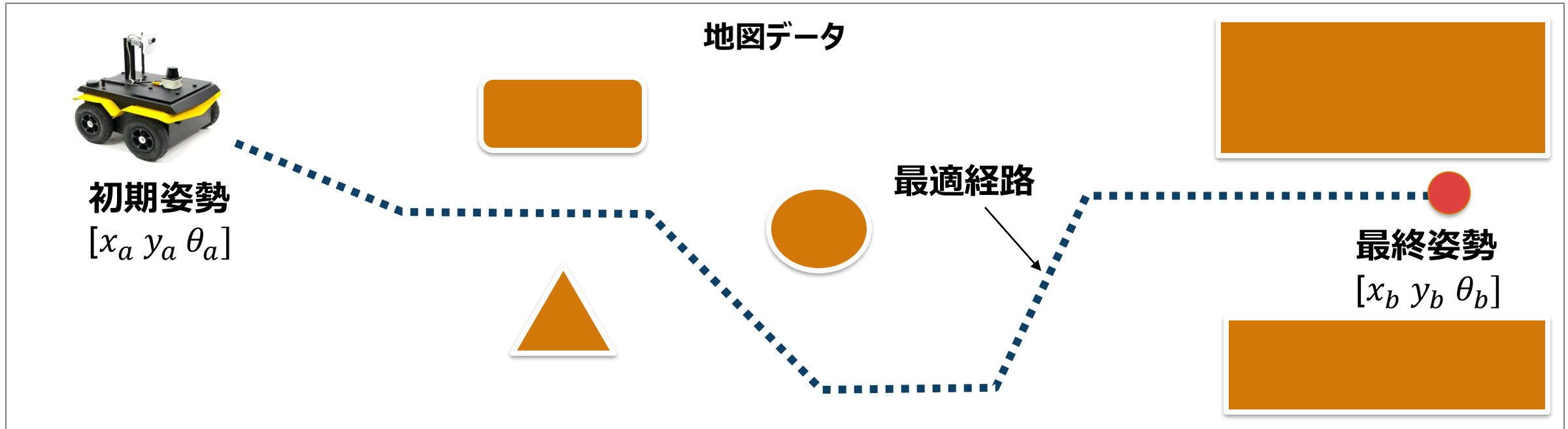


パスプランニングとは？

地図データ(障害物表現)
初期姿勢(自己位置推定)
最終姿勢(目標姿勢)

パスプランニング
アルゴリズム

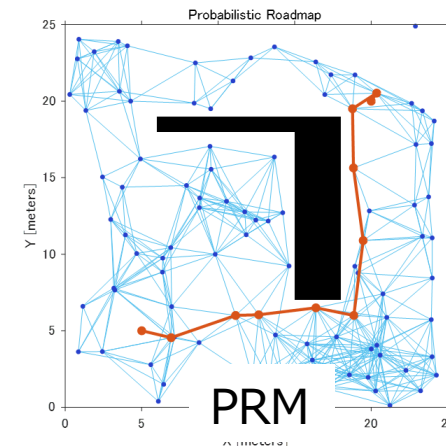
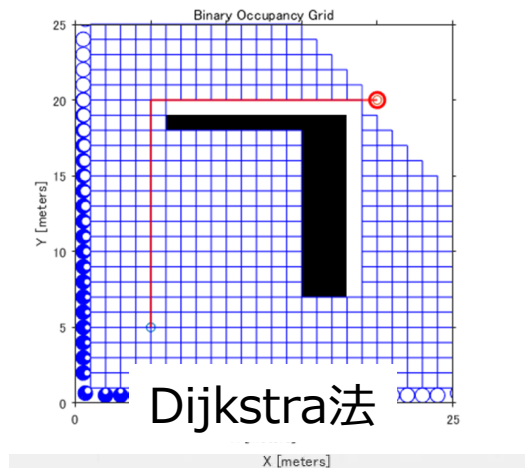
最適経路



複合制約の下、ロボットの最適経路を探索する
アプリケーション：倉庫内搬送、車線変更、追い越し、自動駐車

パスプランニングアルゴリズムの分類

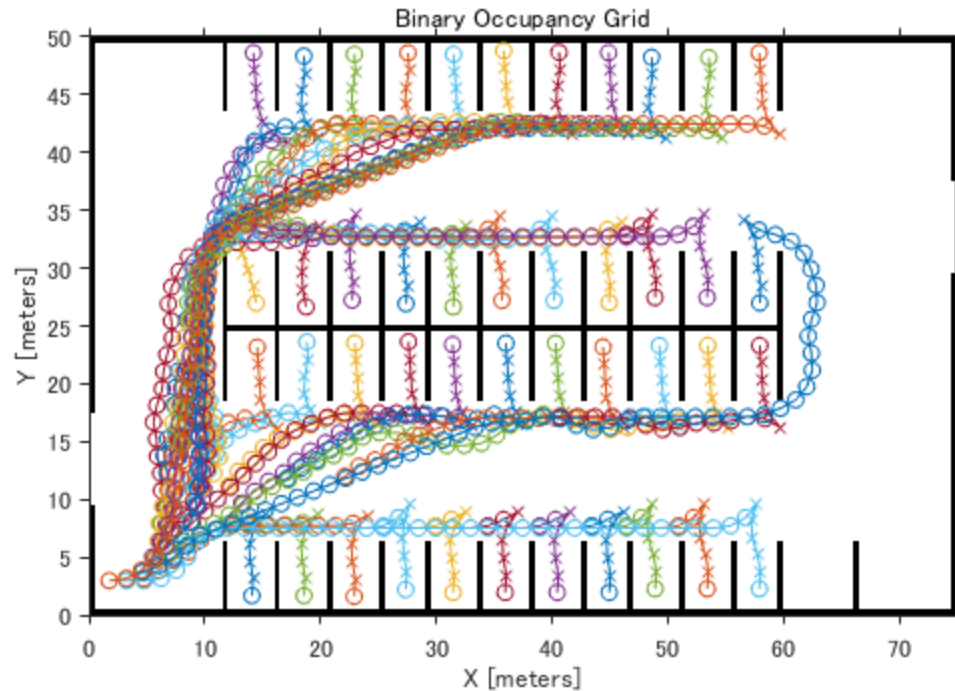
- グリッドベースの探索
 - 探索空間をグリッドで分割
 - カーナビや電車の経路探索など広く利用
 - 探索空間が連続の場合や次元数が高い場合には時間がかかる
- 代表的なアルゴリズム
 - Dijkstra法、A* (A Star)
- サンプルベースの探索
 - 探索空間をランダムにサンプリング
 - サンプリングした点をつなげてパスを作る
 - 探索空間が連続の場合や多次元の場合に有効
- 代表的なアルゴリズム
 - PRM, RRT, RRT*



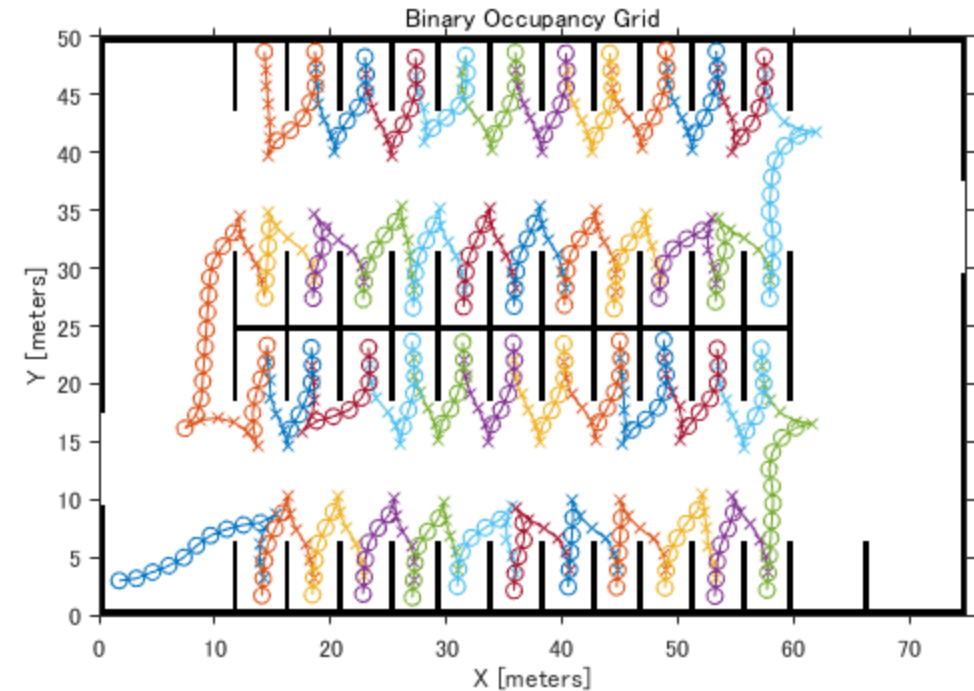
パスプランニング：グリッドベースの独自アルゴリズム実装

- A*アルゴリズム(運動学モデルを使った自動駐車)

スタート地点から各駐車スペースまでの経路



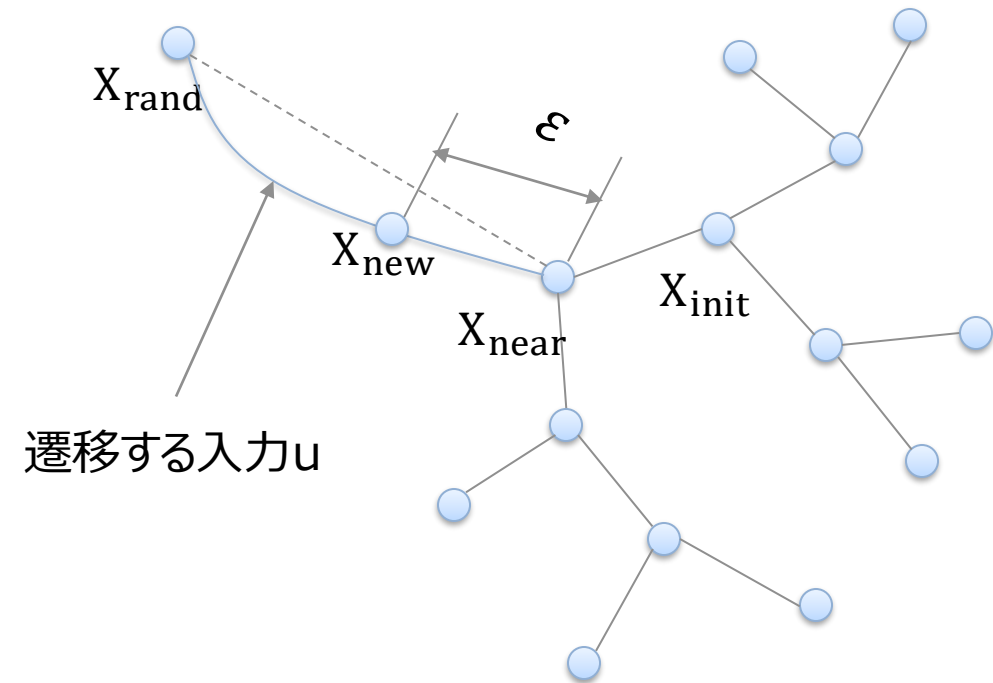
各駐車スペースを周る経路



MATLABのgraphオブジェクトをベースに独自のプランナーを構築することも可能

パスプランニング：ランダムベースのToolbox提供関数

- RRT*アルゴリズム
- サンプリングベースの経路探索手法
 - 初期地点を X_{init} としてグラフを構築
 - 全状態空間からランダムに1サンプルとり、 X_{rand} とする
 - X_{rand} に一番近いノードを X_{near} とする
 - X_{near} から X_{rand} に遷移する入力 u を求める
 - X_{near} から X_{rand} に直線を引き、距離 ϵ の点を X_{new} としてグラフに追加
 - これを繰り返してゴールに到達するまでグラフを拡張していく

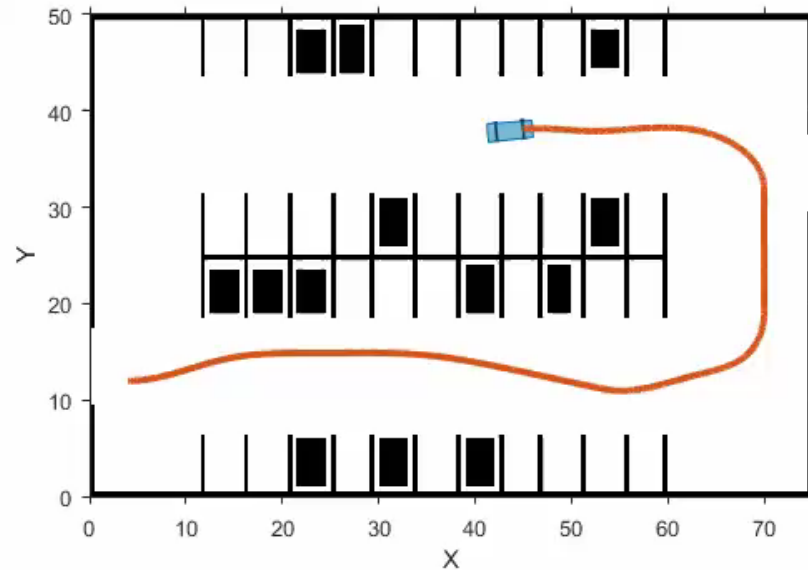


```
planner = pathPlannerRRT;
```

Automated Driving System Toolbox R2018a

パスプランニング : Toolbox提供関数を活用

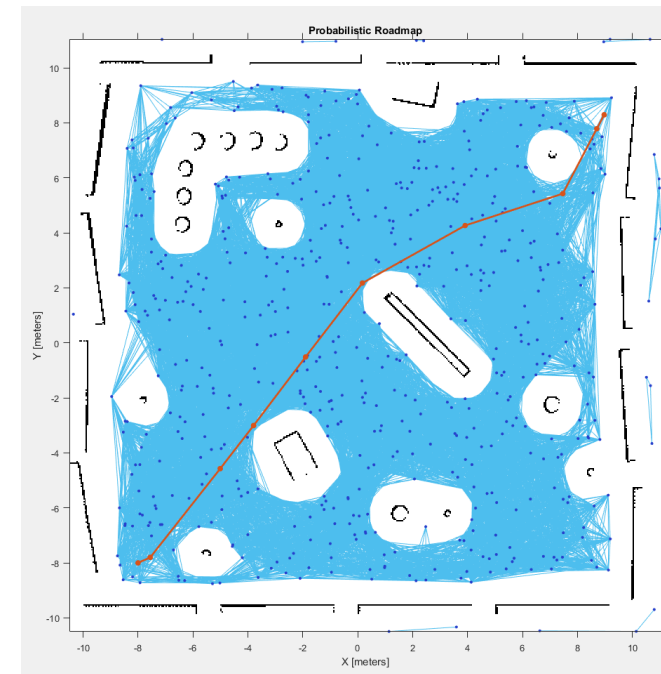
- RRT*(Rapidly exploring Random Trees)



```
planner = pathPlannerRRT;
```

Automated Driving System Toolbox R2018a

- PRM(Probabilistic Roadmap)

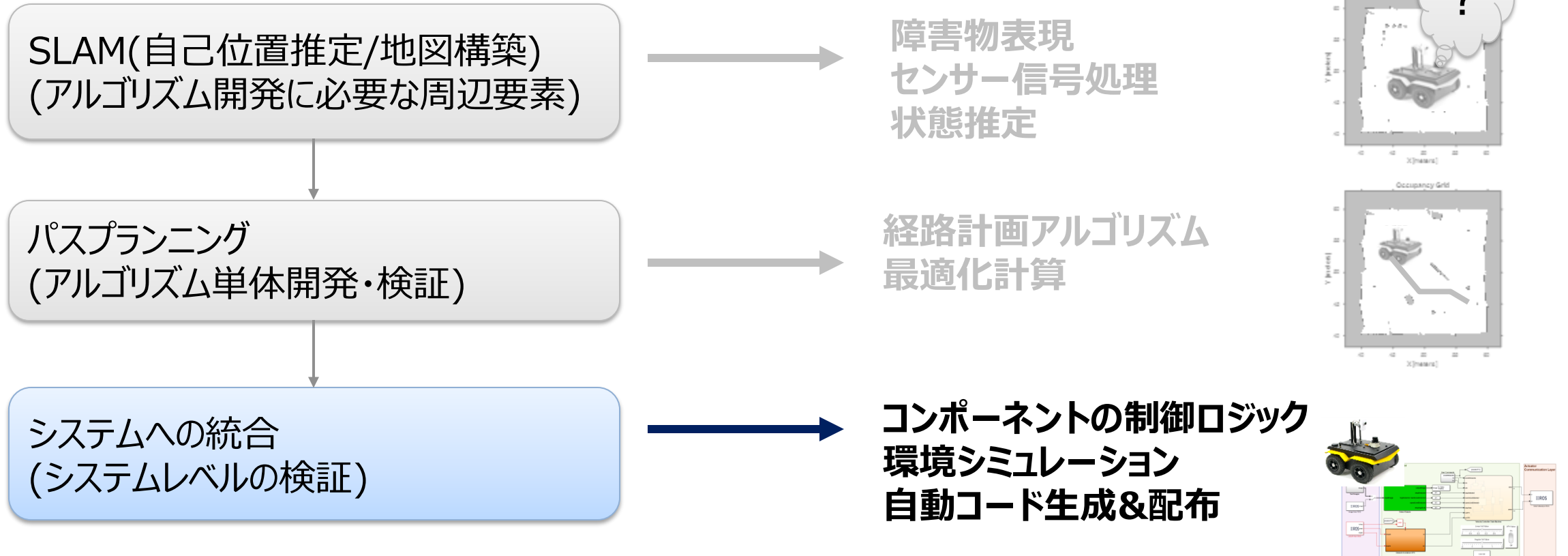


```
planner = robotics.PRM;
```

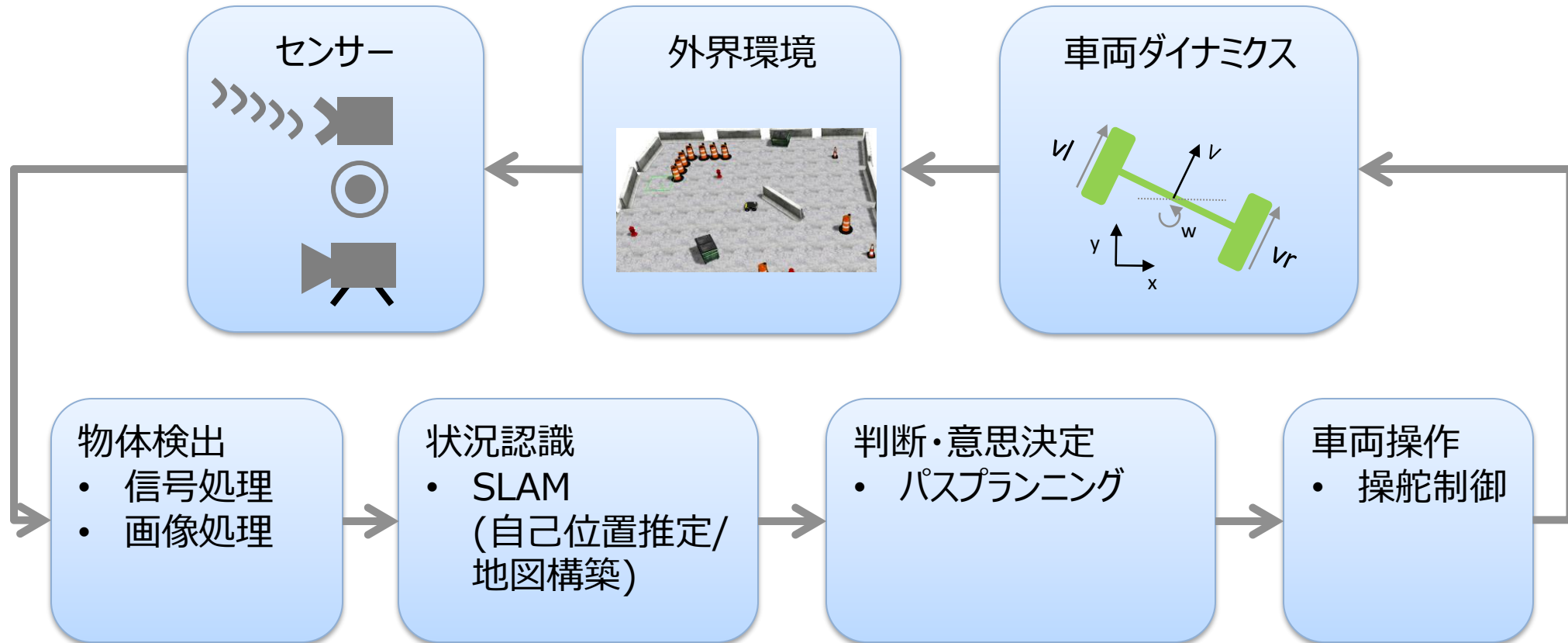
Robotics System Toolbox

Toolbox提供のアルゴリズムを活用してすばやく開発

パスプランニングアルゴリズムの開発フロー



自律移動システムの開発検証



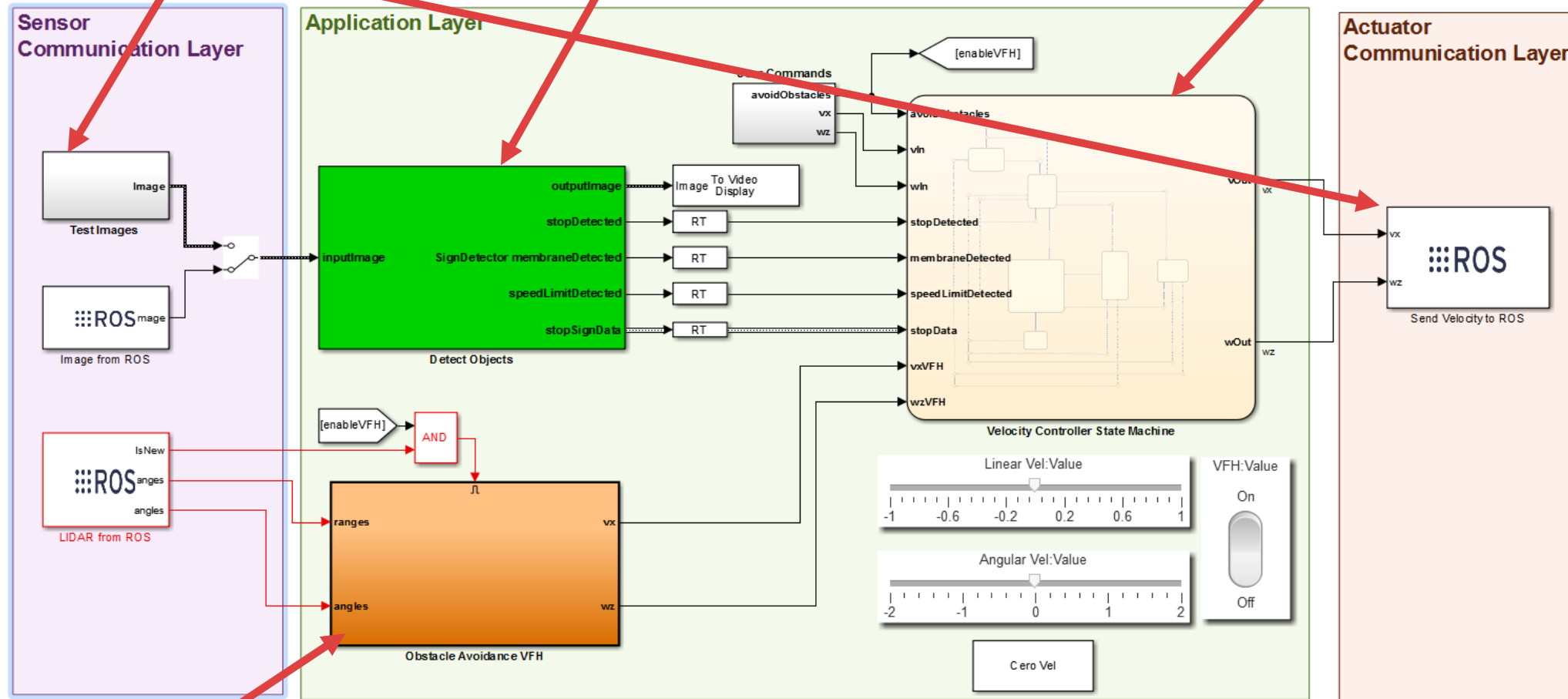
各コンポーネントをどのように統合してシステムレベル検証を行うか？

システムレベル設計検証環境 Simulink

ROSによるデータ交換

物体識別器

パスプランニングアルゴリズムや
ステートマシンによる状態管理



衝突回避

ブロック線図環境で直感的にコンポーネントを配置・統合

ROSによる協調シミュレーションの例(標識認識/障害物回避)

The image displays a ROS simulation environment with two main windows:

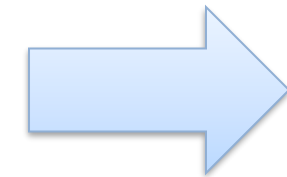
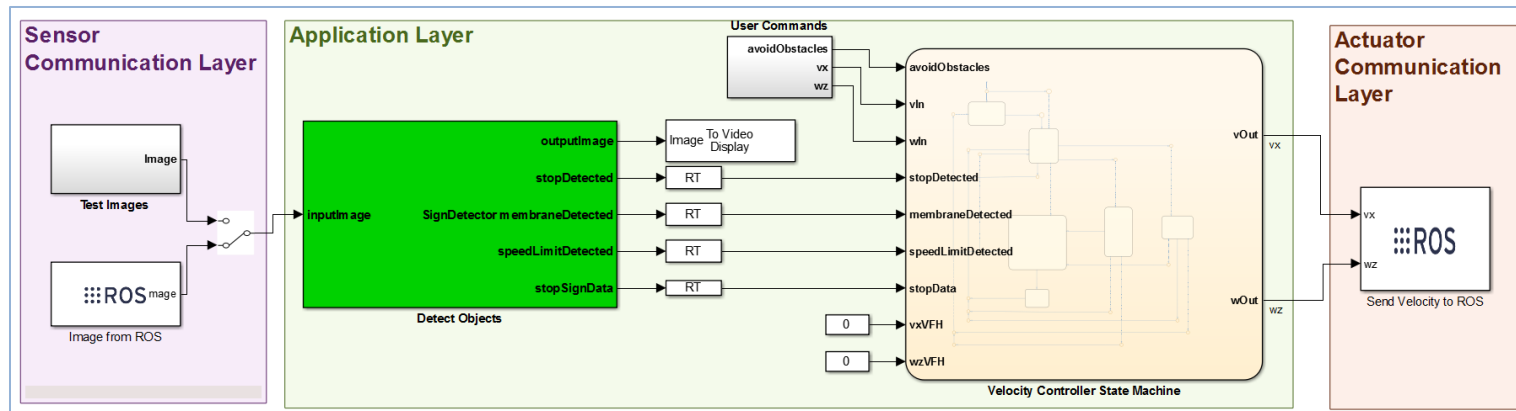
- testSignDetector - Simulink:** This window shows a control system diagram. It includes a 'Detect Objects' block (green) that processes 'inputImage' and outputs 'stopDetected', 'membraneDetected', 'speedLimitDetected', and 'stopSignData'. These signals are processed by a 'Velocity Controller State Machine' (yellow) and an 'Obstacle Avoidance VFH' block (orange). The VFH block outputs 'vsVFH' and 'wsvFH'. The state machine outputs 'vIn' and 'wOut'. Below the diagram are sliders for 'Linear Vel-Value' and 'Angular Vel-Value', and a 'Zero Vel' button.
- Gazebo:** This window shows a 3D simulation of a robot in a virtual environment. The robot is a small yellow and black vehicle. The environment contains concrete barriers, a speed limit sign (60), a warning sign (triangle with exclamation mark), and an orange traffic cone. A red laser line is visible, representing the robot's sensor range.

At the bottom of the screen, there is a taskbar with various application icons and system status indicators, including the time '00:00:11:28.816' and 'FPS: 15'.

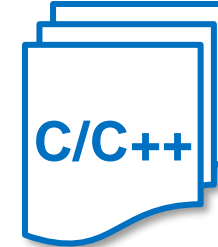
コード生成&実機プロトタイピング

MATLAB Coder™, Simulink Coder™
 Embedded Coder®
 Simulink Report Generator™

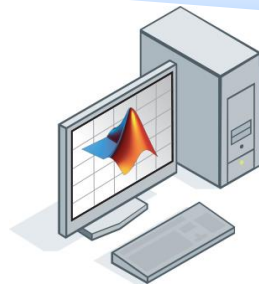
- コード生成機能でROSノードを生成し、ROS対応ロボットに直接実装



ROSノード
(コード生成)



ROSシステムに
ROSノードとして
実装



開発用PC
(MATLAB/Simulink)

作成したモデルを自動コード生成機能を使って容易に実装

まとめ

- SLAM(地図表現と自己位置推定)などのすぐに試せる高度な機能
- 既存機能の組み合わせでパスプランニングアルゴリズムを独自に実装することも可能
- システムレベルシミュレーションによる動作検証とコード生成による実機検証

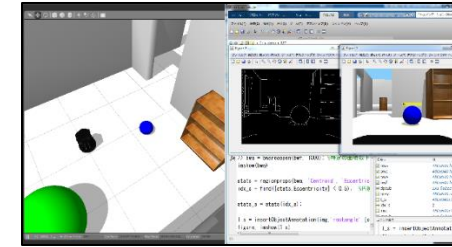


MATLAB/Simulinkを自律移動システムの開発検証にお役立てください！

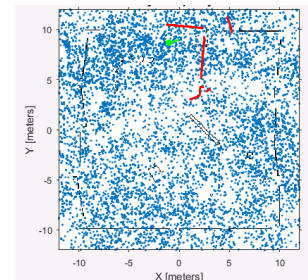
ロボットアルゴリズム開発ソリューション

Robotics System Toolbox

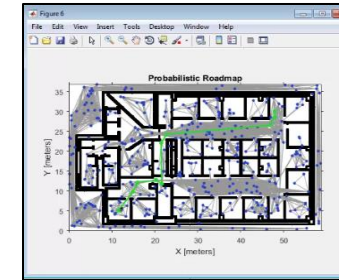
- ROSのインターフェイス提供
 - MATLABをROSマスター、ノードとして起動
 - 直接ROSネットワークに接続して検証
- ROSノード生成
 - SimulinkモデルからC++ ROSノードを生成 (Simulink Coder必須)
- ロボットアルゴリズム開発の支援
 - オイラー角、クォータニオン、座標変換などの便利な関数群
 - SLAM、パスプランニング、VFH+、モンテカルロローカリゼーション(MCL)などの高度な移動用ロボットアルゴリズム
 - ツリー構造表現、逆運動学解析などのマニピュレーターアルゴリズム



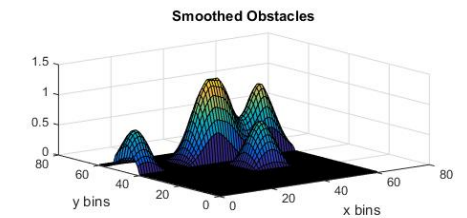
シミュレータや実機とのROS連携



自己位置推定(AMCL)



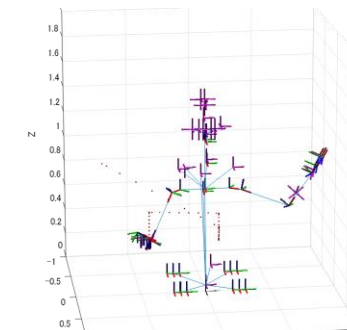
確率的ロードマップ法(PRM)



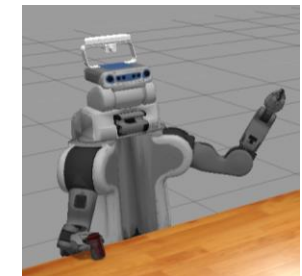
衝突回避(VFH+)



LiDAR-SLAM



ロボットマニピュレーターアルゴリズム開発

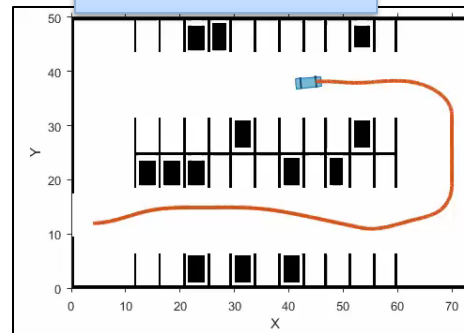


自律移動/ADAS/自動運転向けアルゴリズム開発検証

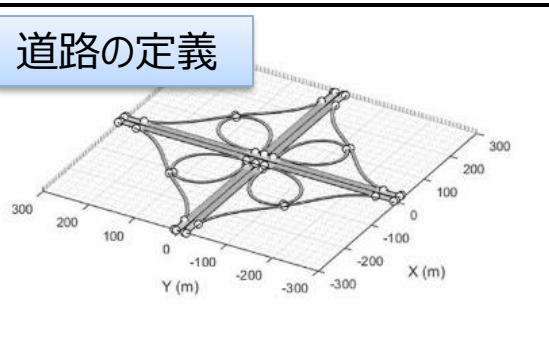
Automated Driving System Toolbox™

- データの可視化
 - 鳥瞰図変換、白線検出・可視化
- 自動車での画像処理・トラッキングに特化したアルゴリズム
 - 拡張カルマンフィルタ、Unscentedカルマンフィルタ
- Ground Truth Labeler
 - 動画に対してラベル付け
- テストシナリオ生成

パスプランニング



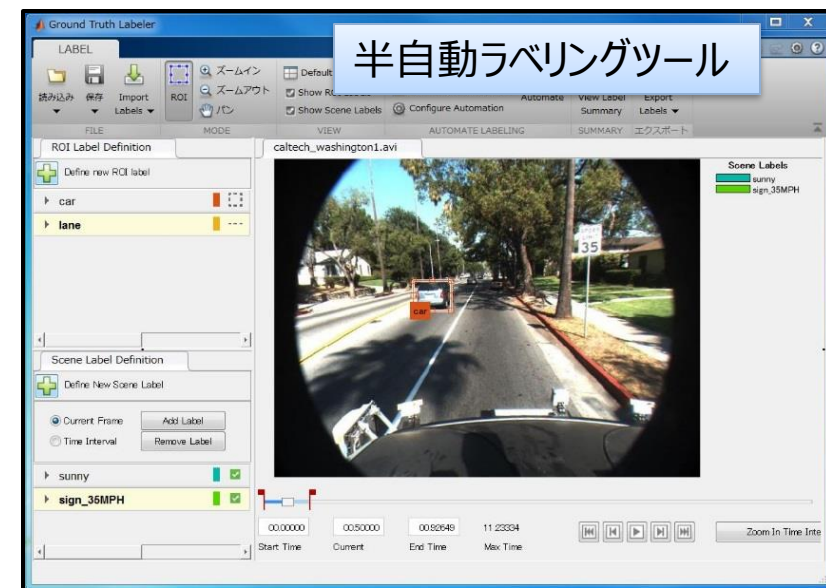
データの可視化機能



各種認識



半自動ラベリングツール





© 2018 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.