version 1.4

TIER IV ACADEMY 自動運転システム構築塾

Day3 Autoware 演習



目次

第1章:データの記録・再生とセンサーキャリブレーション

1. データの記録・生成

2. センサーキャリブレーション

第2章:3次元地図の作成と自己位置推定

- 1. 3次元地図の作成
- 2. 自己位置推定
- 第3章:物体検出と信号検出
 - 1. 物体検出
 - 2. 信号検出
- 第4章:経路生成と経路計画
 - 1. 経路生成
 - 2. 経路計画

第5章:経路追従と車両制御



Autoware 演習

第1章: データの記録・再生とセンサーキャリブレーション 1. データの記録・生成





● ROS ではセンサ等のログデータを ROSBAG と呼ばれるファイル形式で保存できる

ROSBAG を再生することで記録時のセンサ情報を再現しシミュレーションすることができる
 Autoware では GUI で ROSBAG の記録・再生を行う機能を提供している







記録するトピックの配信

ando@ando-Diginnos-PC:~ \$ cd ~/Autoware/ros/ ando@ando-Diginnos-PC:~/Autoware/ros \$./run

- 1. Autoware を起動する
 - A) 端末で以下のように run を起動 (あるいはファイル マネージャから run を起動)
 - \$ cd ~/Autoware/ros/

\$./run

	untime Man	ager	Concerting	lakasfa as	Databasa	cievlation	Chalura	Trains	
UICK Start	GNSS Con	fig]	Computing		Points Filte voxel ring_f distar rando	simulation _grid_filter [s ilter [sys] [ar nce_filter [sys m_filter [sys	status sys] [app op] s] [app]] [app]]	
 Velod Velod Velod Velod 	yne HDL-64e yne HDL-64e yne HDL-32e yne VLP-16	-S2 [config] -S3 [config] [config] [config]				Calibratio Calibratio	on Tool K n Publisi	iit her	
Hoku Hoku SICK L	yo TOP-URG yo 3D-URG .MS511 BL Single					Points Virtual S	s Image can Imag	e	
Other Sen	sors					Scan	Image	Topics Topics I I I I I I I I I I I I I I I I I I I	
0.0%	9.1%	0.0%	27.3% 0	.0% 0	.0% 0.	0% 0.0	1G)% /v /v Cl	RVIZ usr/bin/vncsen usr/bin/X (9.4 ompiz (9.4 %C bin/init (0.0 % threadd] (0.0 6GB/31	RQT ver-x11-core (18.8 %CPU) PU) %CPU) %CPU) GB(19%)
_{cpuo}	cpu1	CPU2 OPU2	CPU3 C	PU4 C	PUS CI	PU6 CP	U7 N	1emory	00(10)0)

- 記録したいデータ(ROSトピック)を
 パブリッシュする ROSノードを ☑して起動
 - 「Velodyne HDL-32e」: Velodyne のデータを /points_rawに配信するノード



データの記録 – 手順 (2/2)

ROSBAG の記録

3. ROSBAG を記録する

- A)「ROSBAG」ボタンをクリックしてダイアログを表示
- B)「Ref」ボタンをクリックして、記録するファイル名を指定
- C) 「Refresh」ボタンをクリック
- D) 記録したいROSトピックを 図する
- E)「Start」ボタンをクリックすると記録開始
- F)「Stop」ボタンをクリックすると記録終了

	autoware-20161011152118.rosbag Ref
●● ◎ Runtime Manager	split size MB
Quick Start Setup Map Sensing Computing Interface Database Simulation Status Topics	Start Stop
Serial GNSS [config]	D mage law
IMU Oring_filter [sys] [app]	/lf stat
Crossbowvg440 O distance_filter [sys] [app] UDARc O random filter [sys] [app]	<pre>/light_color_managed</pre>
Velodyne HDL-64e-52 [config] Calibration Tool Kit	<pre>//ndt_stat</pre>
Velodyne HDL-32e [Confi] Calibration Publisher	/pmap_stat
Velodyne VLP-16 [control] Hokuyo TOP-URG	<pre>/points_raw</pre>
Hokuyo 3D-URG Points Image	/rosout
IBEO 8L Single Virtual Scan Image	/rosout_agg
Other Sensors Scan Irrage	/route_cmd
ROSBAG RVIZ ROT	<pre>/steer_cmd</pre>
0.0% 9.1% 0.0% 27.3% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%	<pre>/to_rtmgr</pre>
compto (54 % CPU) compto (54 % CPU) /sbin/mit (0.0 % CPU) /bbin/mit (0.0 % CPU)	🗌 /topic1
CPU0 CPU1 CPU2 CPU3 CPU4 CPU5 CPU6 CPU7 Memory	1 1 1 c 2
∧utoWare	Refresh



ROSBAG Record

データの再生 – 手順

ROSBAG の再生

- 1. 「Simulation」 タブで bag ファイルを再生
 - A)「Simulation」タブを開く
 - B)「Ref」ボタンをクリックして、再生するbagファイルを指定
 - C)「Play」ボタンをクリックすると再生開始
 - ▶ ROS パラメータ「/use_sim_time」が「true」になる
 - D)「Pause」ボタンのクリックで、一時停止・再生
 - E)「Stop」ボタンをクリックすると再生終了

▶ ROS パラメータ「/use_sim_time」が「false」になる





Autoware 演習

第1章:データの記録・再生とセンサーキャリブレーション

2. センサーキャリブレーション



センサーキャリブレーション - 概要

● 車両に取り付けられた LIDAR と カメラの相対位置関係(x, y ,z, roll, pitch, yaw)を算出

● LIDAR と カメラの位置関係から、両者のデータをフュージョン可能

・カメラ画像上に LIDAR のスキャンデータを投影

キャリブレーションには、チェッカーボードを使用











センサーキャリブレーション - 手順 (1/4)

ツールの起動

今回は /points_raw と /image_raw が記録された Rosbag を用いて、
 キャリブレーションファイルを作成

800	Runtime N	lanager					_			
Quick Sta	rt Setup	Мар	Sensing	Computing	Interface	Database	Simulati	on Sta	tus Topics	
/media	/pdsljp/sso	l/log/cal	lib/for-cal	ib-c0-traffic-c	:1-object.ba	g				Ref
Rate:	Star	t Time (s	5):	🗌 Repeat						
Play	Sto		Pause)						
path: // version: duration: start: S end: Su size: 30 messages compress types: s velo topics: / /can /velo	media/pdsi 2.0 5:22s (322 ep 12 2016 ep 12 2016 0.2 GB 8747 ion: none [6 ensor_msg dyne_msgs camera0/ii nera1/imag odyne_pac	ljp/ssd/l s) 11:24:28 11:29:50 5540/654 js/Image /Velodyi mage_ra ie_raw : kets 220	og/calib/f 0.05 (1473) 0.05 (1473) 0.05 (1473) 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.060 0.050	for-calib-c0-tr 647068.05) 647390.85) 021388200f6f 0804fc9533a0 nsgs : senso : sensor_m : velodyne_m	offic-c1-obj off447d0fcd 0e579e63220 r_msgs/image sgs/velodyr	ect.bag 9c64743] c04ae70566 ge neScan]			
							RO	SBAG	RViz	RQT
0.0%	0.0%	0.0	0%	8.3% 0	.0% 0	.0% (0.0%	0.0%	python (9.0 %C top (9.0 %CPU) /sbin/init (0.0 % [kthreadd] (0.0 [ksoftirqd/0] (0 0GB/3	PU) (CPU) (%CPU) 1.0 %CPU) 1GB(2%)
		are ce	U2 (CPU3 C	PU4 C	PUS C	CPU6	CPU7	Memory	

1. Rosbagの再生・停止

- A) 「Simulation」タブの「Play」ボタンをクリックした後 に「Pause」ボタンをクリック
 - > もし、使用するRosbagに/points_rawではなく、
 /velodyne_pacaketsが含まれていた場合は、P.18の手順に従い、
 Velodyne Driverを起動

- 2. Calibration Toolkitの起動
 - A) 「Sensing」タブの「Calibration Toolkit」をクリック



センサーキャリブレーション – 手順 (2/4)

データの選択

🔊 🗊 🛛 Select Input

Select Input Image Topic

/camera0/ima	ge_raw 👻
Cancel	<u>0</u> K



3. キャリブレーションするデータを選択 A) キャリブレーションを行うカメラのトピックを選択 > ここでは /camera0/image raw を選択

- B) キャリブレーションタイプを選択
 - 今回はカメラとVelodyneの位置関係を計算するので、
 「Camera->Velodyne」を選択





データのグラブとキャリブレーションの実行



移動: ↑,↓,→,←,PgUp,PgDn 回転: a,d,w,s,q,e ポイントサイズ: o(小さく), p(大きく) 背景色変更: b

4. データのグラブ

- A) カメラ画像(左)と LIDAR(右)のデータが 表示されていることを確認し「Grab」ボタン をクリック
- 5. グラブされたデータからチェッカーボード に当たっているポイントの選択
 - A) 右下のウィンドウから、チェッカーボードに 当たっている LIDAR のスキャンをクリック
 - ▶ Grabした数だけ、この作業を行う
- 6. キャリブレーションの実行
 - A) 「Calibrate」ボタンをクリック
 - B) ウィンドウの左側に、キャリブレーションの パラメータが更新されていることを確認





キャリブレーションデータのファイルを保存



a Save Calib			
名前(№): calibration.yam			
7オルダの中に保存(E): 4 mapdsljp machanistrop	 サイズ 最終変更日 2015年12月17 2015年12月03 	フォルダの作成(
易所(<u>P)</u> 名前	v	サイズ	最終変更日
🕽 検索 📄 VectaCam2			2015年12月17日
② 最近開いたファイ… WectaCam			2015年12月03日
≧ odsljp デスクトップ □ ファイル・システム Ξ ssd □ 1.06B ポリューム			
Documents Music Music Pictures Videos Oownloads			
			YML
	++*	22211.0	

7. ファイルを保存

- A) 「Save」ボタンをクリック
- B) ファイル名・保存ディレクトリを選択し、「保存」ボタンをクリック



Autoware 演習

第2章:3次元地図の作成と自己位置推定

1.3次元地図の作成



3次元地図の作成 – 概要

- LIDAR のスキャンデータを用いて、3次元地図を生成
- NDT スキャンマッチングによる自己位置推定を行い、推定された位置にスキャンデータを 追加することで3次元地図を作成
 - ・ただし、地図データが大きくなると 位置推定に時間を要するため、LIDAR の点群データを記録し、 再生しながら生成
- 3次元地図を出力する際には、ボクセルグリッドフィルタでダウンサンプリングしたものを PCDファイルに出力
- 地図が広範囲に渡ると誤差が大きくなる場合もある



相対的な位置を求め、スキャンデータを地図に追加



3次元地図の作成 – 構成





3次元地図の作成 手順 (1/5)

Rosbag の記録



1. LIDAR のスキャンデータを Rosbag に記録

A) Velodyne HDL-32eのドライバ(ROSノード)を起動

「Sensing」タブの「Velodyne HDL-32e」の [config] をクリックし、
 以下を指定する

Autoware/ros/src/sensing/drivers/lidar/packages/velodyne/velodyne_pointc loud/params/32db.yaml

▶ 「Velodyne HDL-32e」を ☑する



- B) 「RViz」ボタンをクリックして RViz を起動してデータを確認
 - [File] [Open Config] で以下の設定ファイルを開く Autoware/ros/src/.config/rviz/ndt_mapping.rviz
 - ▶ RVizの「Global Options」の「Fixed Frame」を「velodyne」に変更
 - ▶ 「Points Raw」が表示されることを確認



3次元地図の作成 手順 (2/5)

Rosbag の記録



- C) 「ROSBAG」をクリックし、ROSBAGを記録
 - 「Ref」ボタンをクリックし、ROSBAGを保存するディレクトリと ファイル名を指定
 - ▶ ROS トピックの「/points_raw」を回する
 - ▶ 「Start」をクリックし、ROSBAGの記録を開始
 - ▶ 「Stop」をクリックし、ROSBAGの記録を終了



3次元地図の作成 手順 (3/5)

3次元地図の作成



2. LIDAR の点群データを再生しながら 3次元地図を生成

- A) 地図作成ノード「ndt_mapping」を起動
 - ▶「Computing」タブの「ndt_mapping」を回して起動する



B) ROSBAGの再生

- 「Sumulation」タブで、「Ref」をクリックし、スキャンデータを記録したROSBAGを選択
- ▶ 「Play」をクリックし、再生を開始

3次元地図の作成 手順 (4/5)



C) RViz の「Global Options」の「Fixed Frame」を「map」に変更



D) 3次元地図が生成されていることを確認



3次元地図の作成 手順 (5/5)

3次元地図の保存



- 3. 地図作成の完了後、PCDファイルに保存
 - A) 「Computing」タブの「ndt_mapping」の [app] をクリック して Config ウィンドウを開く

- <
 Runtime Manage Quick Start Setup Map Sensing Computing Interface Database Simulation Status Topics Localization Mission Planning gnss_localizer ndt_mapping topic:/config/ndt_mapping nmea2tfpose [sys] ndt_localizer Resolution -// Indt_mapping [sys] [app]
 // lazy_ndt_mapping [sys] [app] Step Size 0.1 Trans EPS 14 0.01 ndt_matching [sys] [app] icp_localizer Leaf Size 🔫 icp_matching [sys] [app] Use OpenMP vel pose mux vel_pose_mux [sys] [app] autoware-161004 pcd Ref Detection v cv detector Filter Resolution
 0.2 Original dpm ocv [svs] [app] dpm_ttic [sys] [app] PCD OUTPUT rcnn_node [sys] [app] range fusion [svs] [ann] Close Synchronization ROT o (9.2 %C PH 8.3% 8.3% 0.0% 0.0% 0.0% 0.0% it (0.0 %CPU) ddl (0.0 %CPU) 4GB/31GB(13%) CPU0 CPU1 CPU2 CPU3 CPU4 CPU5 CPU6 CPU7 Memory 🐼 AutoWare
- B) 「Ref」をクリックし、PCDファイルを保存するディレクト リ・ファイル名を指定
- C) 地図データのダウンサンプリングのパラメータ「Filter Resolution」を指定(デフォルトは0.2)を指定し、「PCD OUTPUT」ボタンをクリック
- D) 指定したディレクトリにファイルが存在するか確認



Autoware 演習

第2章:3次元地図の作成と自己位置推定 2. 自己位置推定



自己位置推定 – 概要

- LIDARのスキャンデータと3次元地図のスキャンマッチングにより、自己位置(地図内の 位置・向き)を計算
- RViz の「2D Pose Estimate」、GNSSデータ、座標値入力により、位置推定開始位置を指定
- ボクセルグリッドフィルタにより、ダウンサンプリングしたスキャンデータをマッチング に使用





自己位置推定 – 構成



自己位置推定 - 手順 (1/5)

Simulation time, TF, 車両モデルのセット



- 1. Simulation TimeのON
 - A)「Simulation」タブで位置推定に用いるRosbagを選択し、
 「Play」をクリックし、すぐに「Pause」をクリックし、一時
 停止する
 - ▶ ROS Paramのuse_sim_timeをONにセットするため



- 2. 制御位置とVelodyne取り付け位置のTFのセットと車両モ デルのロード
 - A) 「Setup」タブの「Localizer」で「Velodyne」を選択
 - B) 「Baselink to Localizer」において、車両の制御位置と Velodyneの取り付け位置の位置関係を指定 (x:1.2, y:0, z:2.0, yaw:0, pitch:0, roll:0)し、「TF」をクリック
 - C) 「Vehicle Model」をクリック
 - ファイルは選択せず、空欄にしておけばデフォルトのモデルがロード される



自己位置推定 - 手順 (2/5)

地図データの読み込み

😑 🔍 Runt	ime Mana	ger								
Quick Start S	ietup Ma	P Sensing	Computir	ng Interface	Database	Simulation	Status	Topics		
Point Cloue	d ssd	/moriyama/	map/pointo	:loud_map/bi	n_Laser-0016	i900869.pc	d,/media,	/kitsukawa	/ss (Ref
🗌 Auto Upda	ate 1x1	🗘 Area Lis	t: None							Ref
Vector Ma	p									Ref
TF		edia/kitsuka	wa/ssd/mo	rivama/tf/tf.l	aunch				6	Re
Aap Tools			,,							-
PCD Filter			•							Re
	Lear	Size: 0.2	•							
PCD Binariz	er									Re
						ROSB	AG	RViz	R	QT
0.0% (0.0%	0.0%	0.0%	0.0%	0.0% C	0.0% 0.	.0% / [[[sbin/init (0.0 % kthreadd] (0.0 % ksoftirqd/0] (0. kworker/0:0H] rcu_sched] (0.0	CPU) %CPU) 0 %CPU (0.0 %C %CPU)	U) CPU))
		CPU2	СРИЗ	CPU4	CPU5 C	PU6 CI	PU7 I	4GB/310 Memory	JB(13	5%)
v Auto	vvare	,								

- 3. ポイントクラウド地図の読み込み
 - A)「Map」タブの「Point Cloud」の「Ref」をクリックし、
 地図作成で作成したPCDファイルを選択
 - B) 「Point Cloud」ボタンをクリック
- 4. 地図データの位置を表すTFのセット
 - A) 「Map」タブの「TF」の「Ref」をクリックし、 tf_local.launchを選択
 - B)「TF」ボタンをクリック
 - ▶ world フレームと map フレームの間の TF をパブリッシュする





地図データの読み込み

s 🗆 🔍 Ru	ntime M	lanage	r											
Quick Start	Setup	Мар	Sensing	Computing	Inter	face	Database	Sim	ulation	Status	Topics			
Drivers CAN can_converter can_draw can_listener [config]							Points Filt voxe nng dista rand	er l_grid nicer ince_f om_fi	_filter [<u>s</u> [<u>sys]</u> [ap ilter [<u>sys</u>] lter [sys]	ys] [app]] [app] [[app]]			
Cameras								0	alibratio	on Tool K	üt			
PointGrey Grasshoper 3 (USB1) [Config] PointGrey Generic PointGrey LadyBug 5 [Config] USB Generic IEEE1394 Summer 10 2 22							Calibration Publisher							
							Points Image							
GNSS								1	virtual So	can Imag	le			
 Javad I Serial O 	Delta 3 (GNSS	TTY1)	[config]			Scan Image								
									ROSBA	G	RViz	RQT		
0.0%	0.0%	0	.0%	16.7% (0.0%	0.	0% 9	9.1%	0.0	196 / C P / C	usr/bin/X (8.7 ompiz (8.7 %C ython (8.7 %Cl opt/ros/indigo op (8.7 %CPU)	%CPU) PU) /lib/rviz/rviz (8		
CPU0	CPU1	с	PU2	CPU3 C	PU4	C	PU5 C	PU6	CPI	U7 N	Aemory	00(10%)		
Auto	oWc	are												

5. スキャンデータのダウンサンプリング

A) 「Sensing」タブの [voxel_grid_filter] を 🛛 して起動

ick Start Setup Map Sensing d	Computing Inte	erface	Database	Simulation	n Statu	s Topics		
ivers CAN Can_converter can_draw can_listener [config]		· ·	Points Filt voxel ring dista rande	er I_grid_filter filter [sys] [nce_filter [sy om_filter [sy	(= 6) [ar app] ys] [app ys] [app]			
ameras				Calibra	tion Tool	Kit		
topic : /config/voxel_grid_filter		2	•	Calibrati	ion Publi	ublisher		
				Poin Virtual	ts Image Scan Ima	age		
				Sca	n Image			
				ROSE	AG	RViz	RQT	
0.	Cancel	_	/		0.0%	top (8.8 %CPU) /sbin/init (0.0 % [kthreadd] (0.0 [ksoftirqd/0] (0 [kworker/0:0H	6CPU) %CPU) 10 %CPU)] (0.0 %CPU)	
_	_	_				4GB/31	GB(13%)	

B) [app] をクリックし、表示されるウィンドウで「Voxel Leaf Size」を設定(デフォルトは2.0)し、「OK」をクリック



自己位置推定 - 手順 (4/5)

位置推定

Quick Start	Setup	Мар	Sensing	Computing	Interface	Database	Simulation	Status	Topics	
▼ Localize ▼ gass_	tion biocalizer fix2tfpos nmea2tfp calizer ndt_map azy_ndt_ dt_matc cauzer cp_matcl sse_mux vel_pose_ on tector dpm_ocv dpm_ocv dpm_ocv dpm_ocv dpm_ocv fixen_nod cance_fixe tion	e (sys) pose (si mappin hing (si mux (si (sys) ((sys)	ys] ys] [app] ys] [z] [z] [z] ys] [z] [z] ys] [z] ss] [z] ss	pp] ndt c:/config/ndl nitial Pos GNSS dict Pose OFF © ON or Threshold olution		Mission Pla V lane_plar lane lane lane v freespac astat Motion Plar	nning nner _navi [sys] [z _stop [sys] _stelect [sys] _stelect [sys] _nnavi [sys] nning	scolo species	yaw: 0	
0.0%	0.0%	0.	.0% Ste Trai	p Size —)				0.1	÷
			~		(OK	Cancel			

- 6. 位置推定
 - A) 「Computing」タブの 「ndt_matching」 を 図して起動 する
 - B) [app] をクリックし、表示されるウィンドウで、「Initial Pos」を設定(全て0)
 - C)「OK」をクリック
 - 「Rviz」の「2D Pose Estimate」で位置推定開始位置を指定する こともできる

- 😸 🗆 🗉 Runtime Manager Quick Start Setup Map Sensing Computing Interface Database Simulation Status Topics /home/kitsukawa/log/moriyama_150611/autoware-20150611161340.rosbag.bag Def Start Time (s): 0 Rate: Repeat Play Stop Pause path: /home/kitsukawa/log/moriyama_150611/autoware-20150611161340.rosbag.bag version: 2.0 duration: 3:54s (234s) start: Jun 11 2015 16:22:31.40 (1434007351.40) Jun 11 2015 16:26:25.75 (1434007585.75) end: size: 8.7 GB messages: 5858 compression: none [5858/5858 chunks] types: sensor_msgs/lmage [060021388200F6F0F447d0fcd9c64743] sensor_msgs/PointCloud2 [1158d486dd51d683ce2f1be655c3c181] topics: /image raw 3516 msgs : sensor msgs/Image /velodyne_points 2342 msgs : sensor_msgs/PointCloud2 ROSBAG RViz ROT n/init (0.0 %CPU) 0.0% 0.0% 0.0% 0.0% 0.0% 9 1% 0.0% 0.0% threadd] (0.0 %CPU) softirqd/0] (0.0 %CPU) worker/0:0H] (0.0 %CPU) frcu schedl (0.0 %CPU) 4GB/31GB(13%) CPU5 CPU0 CPU1 CPU2 CPU3 CPU4 CPU6 CPU7 Memory 🐼 AutoWare
- 7. Rosbagの再生
 - A) 「Simulation」タブの「Play」ボタンをクリック



自己位置推定 - 手順 (5/5)

位置推定



- 8. RViz で表示
 - A) RViz を起動し、「File」->「Open Config」から 「Autoware/ros/src/.config/rviz/default.rviz」 を選択
 - B) 地図とスキャンデータが重なって表示されていること を確認



Autoware 演習

第3章:物体検出と信号検出

1. 物体検出



物体検出 – 概要

- カメラ画像中から 車両や歩行者を検出する
- LIDAR の点群情報を利用して 検出物体までの距離を求める
- Reprojection により 検出物体の3次元座標を求める











物体検出 – 手順 (1/4)



- 1. Calibrationファイルを読み込む
 - A) 「Sensing」 タブの「Calibration Publisher」ボタンを クリックする
 - B) ファイルの選択ウィンドウが表示されるので、
 Calibration ファイルを選択する
- 2. Points Image を起動する
 - A) 「Points Image」ボタンをクリックする



物体検出 – 手順 (2/4)



3. dpm_tticの起動



4. range_fusionの起動

Tier IV

Intelligent Vehicle

3. dpm_tticを起動する

A) 「dpm_ttic」をチェックし、表示されるウィンドウで、
 左図のようにパラメータを設定し、「Detection Start」
 ボタンを押す

4. range_fusionを起動する

A)「range_fusion」をチェックし、表示されるウィンドウで、「Start」ボタンを押す

5. klt_trackを起動する



Autoware 演習

物体検出 – 手順 (3/4)



6. opj_reprojを起動する

- A) 「opj_reproj」をチェックし、表示されるウィンドウで、 「Start」ボタンをクリックする
- 7. 検出結果表示用のパネルを追加する
 - A) RVizに移り、「Panels」→「Add new panel」を選択する
 - B) 「Image Viewer Plugin」を選択する
 - C) パネルを適宜見やすい位置や大きさに変更する
 - D) 「ImageViewerPlugin」の「Image Topic: 」を下図の ように選択する



7. 検出結果表示用パネルの追加


物体検出 – 手順 (4/4)



- 8. (A)(B)obj_reprojectionの結果を表示する
- Image: finance with the f

Reset Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options

8. (C)obj_reprojectionの結果を表示する



- 8. obj_reprojectionの結果を表示する
 - A) 「Add」ボタンを押す
 - B) 「BoundingBoxArray」を選択する
 - C) Rviz左のトピックリストに「BoundingBoxArray」が 追加されるので、「Topic」で 「/obj_car/obj_label_bounding_box」を選択する

Autoware 演習

第3章:物体検出と信号検出

2. 信号検出



信号検出 – 概要

- カメラ画像から 信号の色を検出する
- 信号機の座標は 位置推定によって得られる 現在位置と VectorMap の情報から算出する
- 検出結果を 経路計画ノードと連携することで 信号停止・発進が可能







信号検出 – 構成

※一部のトピックは省略





信号検出 - 手順 (1/3)

S 🗢 🗉 Runtime Manager					
Quick Start Setup Map Sensin Drivers CAN can_converter can_draw can_listener [config]	Computing Interface	■ Points Filter ■ Points Filter ■ voxel_grid_filter □ ring_filter [sys] □ distance_filter [□ random_filter [sys]	Status Topics [sys] [app] [app] sys] [app] ys] [app]		
Cameras PointGrey Grasshoper 3 (USB1 PointGrey Generic) [config]	Calibr	ration Tool Kit ation Publisher		
PointGrey LadyBug S [Control USB Generic IEEE1394 Baumer VLG-22		Pc	oints Image	$\overline{\mathbf{n}}$	
GNSS Javad Delta 3 (TTY1) [config] Serial GNSS [config]		Virtu	al Scan Image can Image		
50.0% 0.0% 0.0%	10.01 e calibratio	Rpublisher	OSBAG RViz	ROT	
✓ AutoWare	_moriyama/c	alibration/camera_li	idar_3d/prius/n	ic-1504þ7.yml	Rel
	☑ register TF ☑ publish ext ☑ publish can	between lidar and ca rinsic matrix nera_info OK	amera Cancel		

- 1. Calibrationファイルを読み込む
 - A) 「Sensing」タブの「Calibration Publisher」ボタンを クリック
 - B) ファイルの選択ウィンドウが表示されるので、 Calibrationファイルを選択する



信号検出 - 手順 (2/3)

CO Runtime	Manager							
Quick Start Setu	Map Sensi	ng Computing	Interface	Database	Simulatio	n Status	Topics	
- spin_s	IS LODAL LODAL	-		tane_p	lanner			
rcnn_nc	de [sys] [app]			🗆 la	ane_navi [sy	/s] [app]		
range_f	usion [<u>sys</u>] [ap	P		🗆 li	ane rule [sy	s] [app]		
klt_trac	k [sys] [app]		🙆 feat pr	oi				
kf_trac	k [sys] [app]		topic /con	Fig/adjust v				_
🗆 obj_rep	roj [<u>sys</u>]		topic./con	ng/aujust_x	y			
lidar_detect	ог	_	х —				4	
euclide	an_cluster [<mark>sys</mark>]	[app]	у 🗕 🗕		•		-8	-
obj_fus	ion [sys]							
road wizard								
😽 feat_pr	oj [sys] [app]							
region	tlr sys app	_						
▼ viewers		_						
image -	viewer [svs] [ac	al						
image i	d viewer [svs] [appl						
points	image viewer [sys] [app]						
	image d viewe							
vscan i	mage viewer [s					· · ·		
vscan i	mage d viewer	[svs] [app]			OK	Cancel		
			-	0 P	ure_pursuit	- [2A2] (abb	J	
Synchronization						ROSBAG	RViz	RQT
0.0% 9.1	% 10.0%	20.0%	0.0%	0.0%	9.1%	9.1%	python (18.8 %CF /usr/bin/X (9.4 %	VU) CPU)
							compiz (9.4 %CPI	J)
							python (9.4 %CPU	J)
							8GB/310	GB(27%)
CPU0 CPU	I1 CPU2	CPU3	CPU4	CPU5	CPU6	CPU7	Memory	
🐼 AutoW	are							

2. feat_projの起動



3. region_tlrの起動

2. feat_projを起動する

A) 「Computing」 タブの [feat_proj] を図 して起動

- Calibrationが不十分な場合は、[app] から手動で検出位置 を補正できる
- 3. region_tlrを起動する
 - A) [region_trl] を 🛛 して起動

信号検出 – 手順 (3/3)

File Panels	Неір
8 C) 🛛 Ada	∮ <u>N</u> ew Panel
Del Interact Too	ete Panel Select 🔶 Focus Camera 📼 Measure 🖌 2D Pose Estimate Is
📮 Disekalma	ge 🗶
 ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ <p< td=""><td>ptprjons tetron ound color ■ 48; 48; 48 Harperties Katee Katee Ka</td></p<>	ptprjons tetron ound color ■ 48; 48; 48 Harperties Katee Katee Ka
🕨 🗢 🗹 ritim	e 🗌 🗌
▶ 🖈 TF	
Lasers	raw
Vscanl	Points
points	s_map 🦉
Vector	
vector	
withou	
More Ir	Integrated_viewer
	TrafficLightPlugin
	🔻 📄 jsk rviz plugins
	CancelAction
Add	EmptyServiceCallInterfaceAction
	ObjectFitOperatorAction
	PublishTopic Present Astron
	Record Action Report Command Interface Action
	SelectPointCloudPublishAction
	TabletControllerPanel
	TransformableMarkerOperatorAction
	v 📄 rviz
	r Help Selection
	() Time
	X Tool Properties
l	Description:
	A panel widget to show detections result
	···
Pa	nel Name
	ImageViewerPlugin
	<u>Cancel</u> <u>OK</u>

4. 検出結果表示用パネルの追加

WTierW

4. 検出結果表示用のパネルを追加する

- A) RVizに移り、「Panels」-「Add new panel」を選択する
- B) パネルの一覧が出るので、「Image Viewer Plugin」と
 「Traffic Light Plugin」を選択する
- C) パネルを適宜見やすい位置や大きさに変更する
- D) 「ImageViewerPlugin」の「Image Topic:」を 「/tlr_superimpose_image」に変更
- 5. 信号検出位置の調整
 - A) 画像に表示されている3つの円が信号位置からずれてい るようなら「feat_proj」の[app]から調整する





Autoware 演習

第4章:経路生成と経路計画

1. 経路生成



経路生成 – 概要

- Vector Map の情報や 位置推定によって得られる 位置情報から経路を生成
- Vector Map を用いる場合、道路中心線の情報から 経路を生成
- Vector Map が無い場合、実際に走行した時のセンサーのログデータを用いて生成
- 経路は座標・方向・速度を持ったデータ列としてファイルに出力される





経路生成 – 構成



Vector Mapを用いた経路生成 – 手順 (1/2)

	Manage	r							
ick Start Setup	Мар	Sensing	Computing	Interface	Database	Simulation	Status	Topics	
Point Cloud	/medi	a/ando/ss	d_t/data_nag	oya/map/po	intcloud_m	ap/bin_Laser	-00175	00941.pcd	/me Rel
Auto Update	1x1 ‡	Area List	: None						Re
Vector Map	_t/dat	a_nagoya/	'map/vector_	map/driveo	n_portion.cs	sv,/media/an	do/ssd_t	/data_nag	a/ Re
TF	/media	a/ando/ss	d_t/data_nag	oya/tf/tf.lau	inch				Re
ap Tools									
PCD Filter	Leaf Siz	ze: 0.2							Re
DCD Bioasizes									De
PCD Binarizer							-		Re
						ROSBA	AG	RViz	RQT
7.7% 8.3%	0.	0%	0.0% 0	.0% 7	.7% 0	.0% 0.1	0% // b //	usr/bin/X (9.1 9 op (9.1 %CPU) ython (9.1 %CP sbin/init (0.0 % kthreadd] (0.0 9 5GB/310	6CPU) U) CPU) 6CPU) GB(18%)
CPU0 CPU1	C	PU2	CPU3 C	PU4 C	PU5 C	PU6 CP	107 N	Memory	(,

1. Vector Map の読み込み

- A) 「Map」タブの「Vector Map」の「Ref」をクリックし、 Vector Map ファイルを選択
- B) 「Vector Map」ボタンをクリック

2. Vector Map の位置を表すTFのセット

- A) 「Map」タブの「TF」の「Ref」をクリックし、 Vector Mapの場所に対応する tfファイルを選択
- B) 「TF」ボタンをクリック (これはworldフレームとmapフレームの間のTFをパブ リッシュします)

3. RViz の起動

- A) 「RViz」ボタンをクリック
- B) 「File」->「Open Config」か「Ctrl + O」を押して 「Autoware/ros/src/.config/rviz/default.rviz」 を選択
- C) Vector Map が表示されていることを確認する



Vector Mapを用いた経路生成 – 手順 (2/2)



4. waypoint_clickerの起動



5. RVizの「Publish Point」を用いて経路を作成

4. waypoint_clicker の起動

A) 「Computing」タブで [weaypoint_clicker] を 🛛 して起動

- ▶ [app] から経路の速度や、経路ファイルの保存先を指定できる
- 5. RViz の「Publish Point」を用いて経路を作成
 - A) RViz 上に複数の黄色い点が表示されていることを確認する
 - B) RViz 上部の「Publish Point」をクリックし、RViz 上に表示されている黄色い点をクリック
 - RViz でズームして黄色い点の真上にマウスポインタを置くと
 マークが出てクリックできる
 - C) 同じ車線上の黄色い点をクリックしていくことで、 生成される経路が黄色い線で表示される
 - クリックする黄色い点は、離れた点でも接続可能な 点同士ならば経路が生成される
 - 緑や赤の点が交差点などの分岐点に配置されており、 この点をクリックしないとうまく経路が生成されないことが あるので注意
 - D) 適当なところで「Computing」タブの [waypoint_clicker] の チェックをはずすことで、[app] で指定したディレクトリに経 路ファイルが生成される

Rosbagを用いた経路生成 – 手順 (1/2)

888 Ru	intime Ma	anage	r						
Quick Start	Setup	Мар	Sensing	Computing	Interface	Database	Simulation	Status	5 Topics
▼ Localiza ▼ gnss_l □ f □ r ■ ndt_lc □ r □ l □ v ndt_lc □ r □ l ■ v r □ cp_lo □ v vel_pc	tion localizer ix2tfpose inmea2tfpo calizer ndt_mapp azy_ndt_r ndt_match calizer cp_match ose_mux	e [sys] ose [sy ing [sy nappir ning [sy ing [sy	rs] (app) (sys] (a ys] (app) (s) (app)	<mark>● vel_po</mark> Pose Select ● ndt ma p ○ gnss	velo tching	lane lane lane try Select dt matching AN	_stop [<u>sys</u>] _select [<u>sys</u>]	app] [app] on Mode	e
▼ Detectio	on tector dpm_ocv dpm_ttic rcnn_node range_fusi	[<u>sys]</u> [[<u>sys]</u> [e [<u>sys]</u> ion [sy	app] app] [app] s] [ann]			ОК	Cancel		
Synchroniza	ition						ROSBA	G	RViz RQT
46.2%	9.1% CPU1	8. CI	3% 2U2	0.0% 9 CPU3 C	.1% 0 PU4 C	.0% 0. PU5 CI	.0% 25. PU6 CP	0% // // //	opt/ros/indigo/lib/rviz/rviz (64. /usr/bin/X (27.6 %CPU) oython (18.4 %CPU) compiz (9.2 %CPU) /usr/bin/perl (9.2 %CPU) 10GB/31GB(32%) Memory
🐼 Auto	oWa	re							

2. vel_pose_muxの起動



3. waypoint_saverの起動



- 1. Rosbagを再生し、自己位置推定を行う
 - 自己位置推定の章を参照

2. vel_pose_muxを起動する

- A) 「Computing」タブの [vel_pose_mux] を 🛛 して起動
- [app]では「Simulation Mode」のチェックが外れて おり「Pose Select」と「Velocity Select」の両方で 「ndt_matching」がチェックされていることを確認
- 3. waypoint_saverを起動する
 - A) 「Computing」タブの [waypoint_saver] を 🛛 して起動

Rosbagを用いた経路生成 – 手順 (2/2)



- 4. RViz で/waypoint_saver_marker を表示
 - A) RViz の左のトピックリストの下にある「Add」ボタンを クリック
 - B) 表示されたウィンドウの「By Topic」タブから 「/waypoint_saver_marker」の「MarkerArray」を 選択しクリック
 - C) RViz 上で waypoint を記録しているマーカーが表示され ていることを確認



4. RVizで/waypoint_saver_markerを表示

- 5. waypoint_saver を終了する
 - A) Runtime Manager の「Computing」タブから [waypoint_saver] のチェックをはずすことで 経路の生成を終了する
 - [app] で指定したディレクトリに経路ファイルが 保存される



Autoware 演習

第4章:経路生成と経路計画 2. 経路計画



経路計画 – 概要

- ファイルとして保存されている 経路を読み込む
- 経路の速度を 高精度地図情報などを用いて 最適化する
- 複数の車線があれば、どの車線を通るか選択する





経路計画 – 構成





🛇 🖨 🕒 Runtime Manager							
Quick Start Setup Map Sensing	g Computing	Interface	Database	Simulation	Status	Topics	
Localization			🗆 lane	_stop [sys] [app]		
▼ gnss localizer			🗆 🗆 lane	_select [sys]	[app]		
☐ fix2tfpose [sys]			V Francosco	عمممدام			
nmea2tfpose [sys]	Pose Select	Velo	city Select				-
🔻 ndt_localizer 🛛 🔓	ndt ma	tching 🙆 n	dt matching				
ndt_mapping [sys] [app]			AN		onmode		
lazy_ndt_mapping [sys]	ap gliss		AN.				
✓ ndt_matching [sys] [app]]						
<pre>v icp_localizer</pre>							
icp_matching[sys][app]							
 vel_pose_mux vel_pose_mux [sur] [see 							
Detection	2						
▼ cv detector							
□ dpm_ocy [sys] [app]							
o dpm ttic [sys] [app]							
rcnn_node [sys] [app]			OK	Cancel			
range fusion [svs] [ann]			UK	Cancer			
Synchronization				ROSBA	G	RViz	RQT
46.2% 9.1% 8.3%	0.0% 9	.1% 0	.0% 0.	.0% 25.	0% /op /us pyl con /us	ot/ros/indigo/l sr/bin/X (27.6 % thon (18.4 %CF mpiz (9.2 %CPI sr/bin/perl (9.2	ib/rviz/rviz (64.4 %CPU) PU) U) %CPU)
	CPU3 C				U7 M	10GB/310	SB(32%)
	CF05 C		FUS CI	CP CP	07 M	entory	
2	vel no	nse n	nuxの	記動			

S 🛛 🕄 Runtime									
Quick Start Setup	Map Sensing	Computing	Interface	Databa	ase Simulation	Status	Topics		
Localization	▼ Localization ▼ Mission Planning								
▼ onss localize	r			▼ lan	ne planner				
🛛 🚳 waypoint_loa	der				🗆 lane navi [sv	s] [app]			
Driving morivama	/path/aeon_rvuse	nii ver1 20 20	150916a.cs	Ref	lane rule [sy	app]			
	/pacity accon_ryase	.,	- and a second		lane_stop [sy	s] [app]			
Passing				Ref	lane_select [sys] [app]			
decelerate : 1				e	espace planner				
					🗆 astar_navi [s	ys]			
				ti	on Planning				
				ri	iving_planner				
				- 1	🗆 velocity_set [sys] [app]			
				- 1	path_select [sys			
				- 1	lattice_trajed	tory_gen [sys] [app]		
				- 1	lattice_twist_	convert [sys]		
				'a	ypoint_maker				
				- 1	🖉 waypoint_loa	ider [<mark>sys</mark>]	[<u>app</u>]		
	ок	Cancel		- 1	waypoint_sav	/er [<u>sys</u>] [a	app		
🗌 range_f	usion [sys] [app]		_		waypoint_clic	:ker [<mark>sys</mark>]	[app]		
O kit tran	k [eve] [ann]			¥ wa	vooint followe	•			
Synchronization						ROSBAG	RViz RQT		
54.5% 10.0	% 40.0%	25.0%	0.0%	72.7%	22.2%	27.3%	/usr/bin/X (100.0 %CPU)		
							/usr/bin/perl (37.1 %CPU)		
							/usr/bin/ibus-daemon (18.6 %CP nautilus (18.6 %CPU)		
							10GB/31GB(33%)		
CPU0 CPU	1 CPU2	CPU3	CPU4	CPU5	CPU6	CPU7	Memory		
	are								
	aio								

3. waypoint_saverの起動



1. Rosbag を再生し、自己位置推定を行う ▶ 自己位置推定の章を参照

2. vel_pose_mux を起動する

- A) 「Computing」 タブの [vel_pose_mux] を 🛛 して起動
 - [app] では「Simulation Mode」のチェックが外れており「Pose Select」と「Velocity Select」の両方で「ndt_matching」がチェックされていることを確認
- 3. waypoint_loader で経路を読み込む
 - A) [app] から経路ファイルを指定する
 - ▶ 経路は2つまで指定できるが、1つだけでも起動可能



668 Ru	ntime N	lanage	r							
Quick Start	Setup	Мар	Sensing	Computing	Interface	Database	Simulation	Status	Topics	
 Localiza gnss_l f lane_ru topic:/conf Acceleratio Number of Number of 	tion ocalizer ix2tfpos le ig/lane_i n (m/s^2 Zeros Af Zeros Be	rule 2) () nead =		-	- 1 - 10 - 10	 Mission I lane_p la da <l< td=""><td>Planning lanner ane_navi [<u>sys]</u> ane_rule [<u>sys]</u> ane_stop [<u>sys</u> ane_select [<u>sy</u> ace_planner star_navi [<u>sys</u> Planning</td><td> [app] [app]] [app] /s] [app] §]</td><td></td><td></td></l<>	Planning lanner ane_navi [<u>sys]</u> ane_rule [<u>sys]</u> ane_stop [<u>sys</u> ane_select [<u>sy</u> ace_planner star_navi [<u>sys</u> Planning	[app] [app]] [app] /s] [app] §]		
	[ОК		Cancel		riving v v v v v v v v v v v v v	<pre>planner elocity_set [s ath_select [s ath_select [s attice_traject attice_twist_c int_maker /aypoint_loac /aypoint_save /aypoint_click int_follower</pre>	ys] ory_gen [convert [s er [sys] [c er [sys] [c	sys] (app) sys] (app) (app) (app)	
18.2%	10.0%	1	00.0%	27.3%	0.1%	0.0%	10.0%	0.0%	/opt/ros/india	o/lib/rviz/rviz (10)
CRUD	CPUI	. 1	CPU2	CPU2	CRIM	CRUS	CRILE	CPUZ	/usr/bin/X (93 compiz (18.7 python (9.4 % /usr/bin/vncs 10GB/	6 %CPU) %CPU) CPU) erver-x11-core (9.4 31GB(33%)
	oWc	are	CPU2	CP03	CPU4	CPUS	CPUO	CPU7	memory	

4. lane_ruleの起動

🛛 🖨 🗇 Runtime	Manager							
Quick Start Setup	Map Sensi	ng Computing	Interfac	e Database	Simulatio	on Status	Topics	
▼ Localization ▼ gnss_localize □ fix2tfpo □ nmea2ti ▼ ndt_localizer □ ndt_mai □ lazy_ndi ☞ ndt_mai	r ise [<u>sys]</u> fpose [<u>sys]</u> pping [<u>sys]</u> [ap tching [<u>sys]</u> [ap	₽]] [app] 19]		▼ Mission ▼ lane_r □ l ⊗ l ⊗ l □ l ▼ freesp □ a	Planning olanner ane_navi [s ane_rule [s ane_stop [ane_select bace_plann astar_navi [ys] [app] ys] [app] sys] [app] er sys] [app] er sys]		
 icp_localiz icp_n vel_pose_n vel_point cv_detection cv_detection cv_detection dpm_t dpm_t crnn 	Iane_stop raffic Light Red Lig Green Li opic:/config/lai	cht Lai	ne Change << Lei Right i esult	ft Lane	Currnet Kee Sto	Lane p at 60 p in 5.0	km/h ៣	
range kit tr			ОК	Cancel				
Synchronization						ROSBAG	RViz	RQT
9.1% 16.7	% 9.1%	40.0%	100.0%	0.0%	0.0%	18.2%	/opt/ros/in /usr/bin/X compiz (27. /usr/bin/vn top (18.0 % 10GE	digo/lib/rviz/rviz (99. (45.1 %CPU) 0 %CPU) cserver-x11-core (27. CPU) 3/31GB(33%)
CPU0 CPU	1 CPU2	CPU3	CPU4	CPU5	CPU6	CPU7	Memory	
	are							
		5. lane	sto	pの起	動			

4. lane_ruleを起動する

A) 「lane_rule」を 🛛 して起動

 「Number of Zeros Ahead」と「Number of Zeros Behind」は赤信号で止まる際に、停止線の前後に 速度 0 のwaypoint を置く数を決める

- 5. lane_stopを起動する
 - A) 「lane_stop」を ☑ して起動
 - 「Use traffic light recognition」にチェックをいれると信号認識ノードの結果を受け取って自動で青信号経路と赤信号経路を切り替える



😂 🖨 🗉 Runtime Manager	
Quick Start Setup Map Sensing Computing Inter	ace Database Simulation Status Topics
 Localization gns_localizer fix2tfpose [sys] nmea2tfpose [sys] ndt_localizer ndt_mapping [sucl_fone] lazy_ndt_map lazy_ndt_map icp_matching vel_pose_mux Detection v_ot_detector gm_cov [sys] range_fusion 	 tane_planner lane_nxit [sys] [app] (ane_rule [sys] [app] (ane_stop [sys] [app]) (ane_select [sys] [app]) (sys] [app] (sys] [app] (sys] [app] (sys] [app] (app) (app) (app)
Synchronization OK	Rviz RQT python (B 3 KCM) python (B 3 KCM) rop (B 3 KCM) python (B 3 KCM) rop (B 3 KCM) rop (B 3 KCM) (Boh) (B 1 KCM) rop (B 3 KCM) (Boh) (B 1 KCM) rop (B 3 KCM) (B 1 KCM) rop (B 3 KCM) (B 1 KCM) rop (B 1 KCM) (B
	CPUS CPUG CPU7 Memory

6. lane_selectの起動

🕲 😑 🐵 Runtime Manager							
Quick Start Setup Map Sensing Com	outing Interface	Database Simulation	Status Topics				
Localization		Mission Planning					
v qnss localizer		Iane_planner					
😣 velocity_set		🗆 lane_navi [sys					
Use Crosswalk Detection		Vane_rule [Sys					
topic:/config/velocity_set		■ lane_select [s	ys] [app]				
Others Distance (m)	15	ireespace_planner					
Detection Range (m) —	1.3	🔷 🗧 🗆 astar_navi [<u>sy</u>	<u>s</u>]				
Deceleration Range (m) 📮	0	tion Planning					
Points Threshold	10	velocity set [sys] [app]				
Detection Height Top (m)	0.1	path_select [s	VS				
Detection Height Bottom (m)	-1.8	<pre>lattice_trajectory_gen [sys] [app] lattice_twist_convert [sys] lattice_twist_convert [sys]</pre>					
Deceleration (m/s^2)	1						
Velocity Change Limit (km/h)	15	waypoint_maker	der [svs] [app]				
OK Cance		 waypoint_tout waypoint save 	er [sys] [app]				
range_rusion [sys] [app]		waypoint_clic	ker [sys] [app]				
O bit track four [ann]		V upupoint Follower					
Synchronization		F	RViz RQT				
100.0% 54.5% 40.0% 40.0%	6 0.0%	9.1% 10.0%	30.0% /usr/lib/unity/unity-panel-service /usr/bin/X (92.4 %CPU) /usr/bin/pert (73.9 %CPU) dbus-daemon (9.2 %CPU) compiz (9.2 %CPU)				
		_	8GB/31GB(26%)				
CPU0 CPU1 CPU2 CPU	3 CPU4	CPU5 CPU6	CPU7 Memory				
₩ AutoWare							
7		いつちまし					
7. Velo	OCITY_S6	れの起勤					
0 mot		ようせき					
ö. pat	n_seieo	しした劉					

Intelligent Vehicle

6. lane_selectを起動する

A) [lane_select] を 🛛 して起動

- ▶ [app] から追従する経路を選択できる
- ▶「waypoint_loader」で1つしか経路を読み込んでいない場合は、対応する経路を選択する

7. velocity_setを起動する

- A) [velocity_set] を 🛛 して起動
 - ▶ パラメータは左下図を参照
 - 経路上の物体検出を行うならば、「Sensing」タブの
 「Virtual Scan Image」ボタンをクリックする

8. path_selectを起動する



経路計画 - 手順 (4/4)

- 9. RVizで確認する
 - ▶ 下図のように、薄い緑色の経路(Global Waypoints)と濃い緑色の細い経路(Local Waypoints)が 表示されていれば、ここまでのノードは正常に動いている

▶ 経路上の前方に物体があれば、下図のように赤いボックスが表示され、物体を検出したことを示す





Autoware 演習

第5章:経路追従と車両制御



経路追従と車両制御 – 概要

- 経路を追従できるような車両制御信号の計算を行う
 - 車両制御信号 (v, ω): v...速度、ω...角速度
 - 使用するノード: pure_pursuit、twist_filter
 - subscribeするトピック:/current_pose、/current_velocity、/final_waypoint
 - publishするトピック:/twist_raw、/twist_cmd
- 経路追従と車両制御の流れ
- 1. pure_pursuit $\mathcal{I} \mathbb{k}$
 - ・ 自己位置と経路上の目標点を通る円の曲率を計算
 - ・計算した曲率と現在速度から目標角速度を計算し、 トピックとしてpublish
- 2. twist_filter $\mathcal{I} \mathbb{k}$
 - 目標速度、角速度から想定される横加速度を計算
 - ・横加速度が一定値を超える場合は目標速度を減速し、 トピックとしてpublish
- 3. 車両に送信して制御を行う





経路追従と車両制御 – 構成





経路追従と車両制御 – Rosbagを使用する場合の手順(1/4)

前提

- 自己位置推定が完了し、 /current_pose 、/current_velocity が publish されている
- /final_waypoints が publish されている (path_selectまで起動済)
- ROSBAG は Pause しておく





経路追従と車両制御 – Rosbagを使用する場合の手順 (2/4)

pure_pursuitの設定、及び起動

- 1. 左図、pure_pursuit の [app] をクリックすると右図が出現する図のような設定になっていることを確認
- 2. pure_pursuit を 🛛 して起動

😣 🖨 💷 Runtime Manager		
Quick Start Setup Map Sensing Computing Inter	ace Database Simulation Status Topics	
▼ Localization	<pre>Ø lane_select [sys] [app]</pre>	
▼ gnss_localizer	▼ freespace_planner	
fix2tfpose [sys]	🗆 astar_navi [<u>sys</u>]	
mea2tfpose sys	Motion Planning	
<pre>* ndt_localizer</pre>	driving_planner	
Indt_mapping[sys][app]	፼ velocity_set [<u>sys</u>] [<u>app</u>]	😣 waypoint_follower
<pre>lazy_ndt_mapping[sys] [app]</pre>	፼ path_select [<u>sys</u>]	
✓ Idc_Indc(ing (sys) (app)	<pre>lattice_trajectory_gen [sys] [app]</pre>	Waypoint
\Box ico matching [sys] [app]	<pre>lattice_twist_convert [sys]</pre>	
▼ vel pose mux	waypoint_maker	
vel pose mux [sys] [app]		Velocity 10 🗘
▼ Detection	waypoint_saver [sys] [app]	
cv_detector	▼ waypoint follower	Lookahead Distance — 4
dpm_ocv [sys] [app]	ure pursuit [sys] [app]	lookabead ratio
dpm_ttic [sys] [app]	twist_filter [sys] [app]	
rcnn_node [sys] [app]	wf_simulator [sys] [app]	minimum_lookahead_distance — 🛛 🗍 4
Synchronization	ROSBAG RViz RQT	
18.2% 0.0% 0.0% 10.0% 54.5%	20.0% 10.0% 0.0% /opt/ros/indigo/lb/rviz/rviz (82 compiz (27.3 %CPU)	relative_angle_threshold
	top (9.1%CPU) /sbin/init (0.0%CPU)	👿 Linear Interpolate
	12GB/31GB(39%)	
	cros cros memory	OK Cancel

経路追従と車両制御 – Rosbagを使用する場合の手順 (3/4)

twist_filter の設定、及び起動

1. 左図、twist_filter の [app] をクリックすると右図が出現する 図のような設定になっていることを確認

2. twist_filter を 🛛 して起動



経路追従と車両制御 – Rosbagを使用する場合の手順 (4/4) ROSBAG の Pause を解除すれば、下図のようになる





経路追従と車両制御 – トピックの確認(参考)

トピックが publishされているかを確認するには...

- 「Topic」 タブで「Echo」 を ☑ する
- 2. /[topic_name]をクリック、ここでは /twist_cmd を表示
 - ・もし存在しない場合は Refreshをクリック

上記が満たすと、 <u>青で囲った場所にトピックが流れる</u>

😣 🖻 🗊 Runtime Manager							
Quick Start Setup Map	Sensing Computing	Interface	Database	Simulation	Status	Topics	
/config/pedestrian_dpm			Echo				
/config/pedestrian fusion /config/random filter /config/ring filter /config/ring filter /config/ving filter /config/vist_filter /config/velocity_set /config/velocity_set /config/waypoint_follower /from rtmgr /light_color managed /ndt_stat /rosout_agg /steer_cmd /to_rtmgr /to_rtmgr /topic1		lir x y z ar x y y z linf 191 Pul */ */ */ */	near: :: 11.1111111 :: 0.0 :: 0.0 :: 0.0 :: 0.0 :: -0.01298051 o pe: geometry blishers: runime_mana twist_filter (h bsscribers: rostopic_290	111 77543 _msgs/ Iwiststa ager_10347_14 http://LapisLaz 56_147597107	ampeo 7596768 culi2:406 6495 (ht	7021 (hti 92/) tp://Lapi	:p://LapisLa sLazuli2:33
<u>/twist_cmd</u>							
Refresh				ROSBAG	F	Viz	RQT
16.7% 83.3% 0.0	% 0.0% 9.	1% 10	0.0% 0.	0% 0.0%	6 pyti /op con top /usi	non (101.0 % t/ros/indigo npiz (18.4 % (18.4 %CPU /bin/X (9.2 %	CPU) /lib/rviz/rviz (101. CPU) 6CPU)
			PUIS CE	PU6 CPU	7 M	12GB/31	GB(39%)
√ AutoWare				cro			



経路追従と車両制御 – wf_simulator を使用する場合

wf_simulatorとは?

- 車両制御信号 (v, ω) を受け取り、理想的な自己位置及び速度を
 シミュレート
- ・計算式は以下の通り

$$x_{i+1} = x_i + \nu \cos \theta_i \Delta t$$

$$y_{i+1} = y_i + \nu \sin \theta_i \Delta t$$

$$\theta_{i+1} = \theta_i + \omega \Delta t$$

$$i : 0, 1, ..., n$$



経路追従と車両制御 – wf_simulator を使用する場合

構成





Autoware 演習

経路追従と車両制御 - wf_simulator を使用する場合の手順 (1/6)

前提

- 地図、及び /base_waypoints が publish されている (lane_select まで起動済)
- この操作の前にROSBAGを使った作業をしていた場合は、ROSBAGの再生を Stop した後に TFとRViz をそれぞれ再起動しておく
- 「Setup」 タブの「Vehicle Model」で /\$HOME/Autoware/ros/.config/model/sim_default.urdfをロードしておく

8-0 R.	untime Ma	inager							
Quick Start	Setup	Map Ser	sing Comput	ing Interface	Database	Simulation	Status	Topics	
Localizer Velodyn	e 🔿 Hoku	iyo 3D URG							
Baselink to	Localizer								
TF	rospar x: 1.2	am: ros	oaram : rospi	aram : rospar yaw: (am:tf ros	param : tf_j ch: 0 🗘	rospara roll: 0	m : tf	
Vehicle Mod	del								Pef
Venicie Mi	odec								Kei
Vehicle Inf	0								
Vehicle In	nfo								Ref
						ROSBA	G	RViz	RQT
0.0%	0.0%	0.0%	0.0%	53.8%	0.0% 9	.1% 0.0	1% P) /u /s [k [k	/thon (53.0 %C sr/bin/X (8.8 % bin/init (0.0 % threadd] (0.0 % softirgd/0] (0.	PU) 6CPU) CPU) %CPU) 0 %CPU)
CDUO	CDU1	CDU2	CDU2	CDUA	DUE C		17 1	4GB/310	GB(13%)
			CP03	CP04 (LPU5 C	PUO CP	07 IV	lenory	
🛶 /\ur	ovva	e							





経路追従と車両制御 - wf_simulator を使用する場合の手順 (2/6)

wf_simulatorの起動

- 1. 左図、「wf_simulator」の [app] をクリックすると右図が出現する 図のような設定になっていることを確認
- 2. 「wf_simulator」を ☑ して起動





経路追従と車両制御 - wf_simulator を使用する場合の手順 (3/6)

wf_simulatorの起動

- 1. RVizにて、2D Pose Estimateを使用し、初期位置を決定
- 2. 自動車のモデルが現れる、位置が正しくない場合はもう一度 1.を繰り返す





経路追従と車両制御 - wf_simulator を使用する場合の手順 (4/6)

経路の設定

- 1. 「velocity_set」、「path_select」を 🛛 して起動
- 2. 「vel_pose_mux」の [app] をクリックすると 右図が出現する Simulation Mode に ☑する
- 3. vel_pose_mux に ☑ する

😣 🖻 💷 Runtime Manager		
Quick Start Setup Map Sensing Computing Interf	ace Database Simulation Status Topics	
 Localization gnss_localizer fix2tfpose [sys] nmea2tfpose [sys] ndt_localizer 	 ✓ Iane_select [sys] [app] ✓ freespace_planner □ astar_navi [sys] ✓ Motion Planning ✓ driving_planner Pose Select Velocity Select Imatching Imat	
<pre> ndt_mapping [sys] [app] lazy_ndt_mapping [sys] [app] ndt_matching [sys] [app] vicp_localizer icp_matching [sys] [app] vel pose mux </pre>	<pre></pre>	
vel_pose_mux[sys[app] Detection vel_pose_mux desc sample v_detector dpm_ocv[sys][app] dpm_ttic[sys][app] cron_node[sys][app] crane_fusion[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][app] crane[sys][sys][sys][sys][sys][sys][sys][sys	 waypoint_loster [sys] [app] waypoint_clicker [sys] [app] waypoint_follower pure_pursuit [sys] [app] twist_filter [sys] [app] w f_simulator [sys] [app] 	
16.7% 45.5% 54.5% 10.0% 9.1% CPU0 CPU1 CPU2 CPU3 CPU4	9.1% 10.0% 0.0% /opt/ros/indigo/lib/rviz/197.4 /usr/bin/K1.7% CPU) compic (17.7% CPU) /home/h.ohta/Autoware/ros/dev python (8.9% CPU) 10GK31CB(32%)	
∧uto\//are		

経路追従と車両制御 - wf_simulator を使用する場合の手順 (5/6)







Autoware 演習
経路追従と車両制御 - wf_simulator を使用する場合の手順 (6/6)

経路追従の開始

- 1. 「pure_pursuit」、「twist_filter」 を起動
- 2. 下図のように目標点、円弧が表示され、シミュレータが動き出す







Intelligent Vehicle

www.tier4.jp