



2017年11月28日(火) RTミドルウェア講習会@人工知能学会



## 第3部:RTシステム構築実習

#### 国立研究開発法人産業技術総合研究所 ロボットイノベーション研究センター ロボットソフトウェアプラットフォーム研究チーム 高橋 三郎







- 1. 事前確認
- 2. 実習概要
- 3. 実習A
- 4. 実習B
- 5. 応用課題





## 1. 事前確認







- スライド
  - 現在見せているスライドと同じもの
- WEBページ
  - 手順を記載したページ
    - チュートリアル(AIツール、第3部、「データ収集・蓄積」の実習)\_ OpenRTM-aist.htm
    - チュートリアル(AIツール、第3部、「推論結果の検証」の実習) \_ OpenRTM-aist.htm
- OpenRTM-aistインストーラ(Windows用)
  - OpenRTM-aistのインストールに問題があった場合に使用してください
- Pythonパッケージ (Chainer, OpenCV-Python)
  - この講習で使うPythonパッケージです。事前にインストールされていない方は下記を解凍し、 作成されたフォルダに移動後、下記コマンドでインストールして下さい。
    - python setup.py install
- EXE
  - RaspberryPiMouseSimulatorComp.exe(シミュレータ)
- sample
  - 本実習で作成するコンポーネントの見本です
    - NameToVelocity
    - ImageDataCollector





<u>インストールの確認</u>

- OpenRTM-aist 1.1.2
- Python
- PyYAML
- Doxygen
- Python用エディタ
- Python パッケージ
  - Chainer
  - OpenCV-Python





# 2. 実習概要





ヨ標

- RTM (OpenRTM-aist) を利用し、人工知 能技術を応用したロボットシステムを構築します
- 深層学習による画像認識を利用した移動ロボット制御システムを作成することで、実際の研究、開発へのアプリケーション応用について学びます









- ロボットを利用するか?
  - 既存のロボット制御用コンポーネントが利用できる
- 修正を行いながらの繰り返し開発を行うか?
   コンポーネント分割により修正を局所化できる
- 第三者が設計情報を再利用するか?
   コンポーネント単位で疎結合なため再利用性が高い





## <u>RTMの特長(AIツール観点)</u>

- 設計情報の再利用性向上
  - 学習, ロボット制御, データ収集などのロジックを別コン ポーネント化することでシステム変更, 流用時の修正を 局所化
- 分散制御による拡張性向上
  - 制御対象ロボットの追加や出力信号の多重化が容易
  - 制御周期や通信バッファ方式を変更することで, データ 流量や制御タイミングを設定可能

AIST



### AIツールとしてRTMを利用する開発の流れ

$\wedge$	要求定義	<ul> <li>システム要求定義</li> <li>一開発の目的,要求事項・対象データ明確化</li> </ul>
	設計・開発	<ul> <li>システム設計・開発</li> <li>→ システム構成の検討,学習,推論エンジン開発</li> </ul>
	データ収集	<ul> <li>データ収集・蓄積</li> <li>☆RTMを活用できる工程</li> <li>実習: 3-B</li> <li>環境構築,実データの取得</li> </ul>
	学習	<ul> <li>学習(データ前処理含む)</li> <li>データ前処理,学習処理の実行</li> </ul>
	検証	<ul> <li></li></ul>























# 3. 実習A 「推論結果の検証」

NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)





【実習3-A】「推論結果の検証」実習







【実習3-A】「推論結果の検証」実習

:#Python27¥python.exe	
-11-27 18:06:18,631 ImageToObjectPrediction	1 INFO Recognized Object: wardrobe
-11-27 18:06:19,493 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:21,243 ImageToObjectPrediction	INFO Recognized Object: Wardrobe
-11-27 18:06:22,994 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:30,802 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:31,506 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:36,769 ImageToObjectPrediction	INFU Recognized Object: wardrobe
-11-27 18:06:40, 262 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:40,992 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:41,713 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:43.068 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:47,413 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:48,135 ImageToObjectPrediction	INFO Recognized Object: wardrobe
-11-27 18:06:52,677 ImageToObjectPrediction	INFO Recognized Object: sliding door
-11-27 18:06:50,934 IMageToObjectPrediction -11-27 18:06:57 642 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:06:58,410 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:06:59,144 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:06:59,848 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:01 263 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:01.982 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:02,690 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:03,405 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:04.144 IMageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:05,551 [mageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:06,247 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:06,976 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:08 437 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:09, 141 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:09,845 [mageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:10,536 ImageToObjectPrediction	n INFO Recognized Object: pinwheel
-11-27 18:07:11 979 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:12.670 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:13,368 [mageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:14,082 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:15 490 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:16, 197 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:16,895 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-2/ 18:0/:1/,595 ImageloUbjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:18 987 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:19,733 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:20,509 [mageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:21,246 ImageToObjectPrediction	INFU Recognized Ubject: pinwheel
-11-27 18:07:22,673 [mageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:23.381 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:24,102 ImageToObjectPrediction	1 INFO Recognized Object: pinwheel
-11-27 18:07:25 505 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:26,203 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:26.928 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:28 365 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:29.085 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:29,832 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:30,536 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:31 992 ImageToObjectPrediction	INFO Recognized Object, pinwheel
-11-27 18:07:32.686 [mageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:41,148 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:41,862 ImageloObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:43.278 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:43,990 ImageToObjectPrediction	INFO Recognized Object: pinwheel
-11-27 18:07:44,696 ImageToObjectPrediction	1 INFO Recognized Object: pinwheel
-11-27 18:07:46 151 ImageToObjectPrediction	INFO Recognized Object, pinwheel
-11-27 18:07:46, 897 ImageToObjectPrediction	1 INFO Recognized Object: pinwheel
-11-27 18:07:48,273 [mageToObjectPrediction	1 INFO Recognized Object: pinwheel
-11-27 18:07:49,016 ImageloUbjectPrediction	INFO Recognized Ubject: pinwheel
-11-27 18:07:50 438 [mageToObjectPrediction	INFO Recognized Object: pinwheel







### 【実習3-A】物体認識コンポーネント (ImageToObjectPrediction)の仕様

- CNN (GoogLeNet) による物体認識コンポーネント
- 実装は Chainer の ImageNet サンプルを利用
- 認識物体名はImageToObjectPrediction/labels.txtに記載
- 入力された画像に対し、学習済みのモデルで物体名を推論
- 本実習では時間の都合上,
   既に作成済みのものを再利用します



AIST



- 【実習3-A】作成するRTC(NameToVelocity)の仕様
- カメラ画像から認識した物体名が文字列(TimedString型)として入力されます
- 認識される物体名はImageToObjectPrediction/label.txt記載
  - 認識率が比較的高いのは「bow tie」「hook」「pinwheel」「enveloop」でしたので、テスト用に印刷した画像を配布します
- 物体名を速度情報(TimedVelocity2D型)に読み替えてロボットを制御する
  - 物体名を移動方向に割り当てます 前進,右旋回,左旋回,後進
  - 割り当てていない物体名を認識した 場合は停止します









- 学習済みの物体は下記リストに記載されています(1000種類)
   ImageToObjectPrediction/labels.txt
- サンプルコードは比較的認識しやすい下記の物体4種を使っています
   iPod, cellular telehone, suit なども認識しやすいので試してみて下さい















NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)







- RTC Builderによるソースコード等のひな型の作成
- ソースコードの編集
   NameToVelocity.pyの作成
- RTシステムエディタによるRTシステム作成、動作確認
   RTシステム作成
   データポート接続
- Raspberry Piマウスシミュレータとの接続、動作確認
- Raspberry Piマウス実機との接続、動作確認





【実習3-A】 プロジェクト作成

 NameToVelocityコンポーネントのスケルトンコ ードをRTCBuilderで作成する

🗢 RTC Bui	lder - I
ファイル(F) 編集(E) ソース(S) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(P)	実行(R)
🖆 ▼ 🗒 🕼 📥 🛃 🕸 ▼ 🗿 ▼ 💁 ▼ 🍅 🖋 ▼   ½ ▼ 🖏 ▼ 🔶 ▼ →	•
増パッケージ・エクス 8 OA New RtcBuilder Editor	
> 🗁 AbsFunction	
▷ 🖻 AccAndMagnS 🗧 このアイコンをクリック	
> 🗁 AccelerationSe	
> 🗁 AccelerationSe	
Adafruit_PWMs	
> 🗁 ADC_Edison	
ADC_MCP3002	
▷ ADC_MCP3208	
ARToolKitPlus	
Backlash	
> 🗁 Bias	

RT-Component Builder Project		inter All and a second s		×
プロジェクト名(ℙ): NameToVelocity ▼ デフォルト・ロケーションの使用(型) ロケーション(L): G¥workspace¥NameToVelocity			参照( <u>R</u> )	
?	終了( <u>F</u> )		キャンセ	١Ļ





### 【実習3-A】 基本プロファイルの入力

プロパティ	内容
モジュール名	NameToVelocity
モジュール概要	任意 (Convert to robot velocity from object name)
バージョン	任意(1.0.0)
ベンダ名	任意
モジュールカテゴリ	任意(Converter)
コンポーネント型	STATIC
アクティビティ型	PERIODIC
コンポーネントの種類	DataFlow
最大インスタンス数	1
実行型	PeriodicExecutionContext
実行周期	1000.0
概要	任意







- 以下のアクティビティを有効にする
  - onInitialize
  - onActivated
  - onDeactivated
  - onExecute
- Documentationは適当に 書いておいてください
  - 空白でも大丈夫です

▼ アクティビティ		
このセクションでは使用する	るアクションコールバックを指定します。	
	コンポーネントの初期化と終了処理	に関するアクション
onInitialize	onFinalize	
	実行コンテキストの起動と停止に	関するアクション
onStartup	onShutdown	
	alive状態でのコンポーネン	トアクション
onActivated	onDeactivated	onAborting
onError	onReset	
	Dataflow型コンポーネント	のアクション
onExecute	onStateUpdate	onRateChanged
	FSM型コンポーネントのフ	アクション
onAction		
	Mode型コンポーネントの	アクション
onModeChanged		





### 【実習3-A】データポートの設定

- 以下のOutPortを設定する
  - ポート名:out
  - データ型: RTC::TimedVelocity2D
  - 他の項目は任意
- 以下のInPortを設定する
  - ポート名:in
  - データ : RTC::TimedString
  - 他の項目は任意

(P) 実行(R) ウィンドウ(W) ヘルプ(H)		- o ×						
		📰 🛃 Java 🔤 RT System Editor 🏠 RTC Builde						
**NameToWelocity 22		° C						
7-34-1								
■ DataPort/20/7/18		• 01						
このセクションではRTコンボーネントのCostaPointテータボート)の情報を設定します。	テータボート: RTコンボーネント聞でテータをやり取りするためのボートです。 ゲータを出力するOurPontと、データを入力するInPontがあります。							
Add TE-1-6 (OutPort)	Add	InPort2OutPortを接続するには、同者のデーク型が同一である必要があります。						
Delete	Delete	InPort: RTコンボーネントにデーダを入力するためのボートです。 他のRTコンボーネントのOutPortと課題され、データを受け取ります。						
		OutPort: RTコンポーネントからデータモ出力するためのポートです。						
		他のRTコンボーネントのinPortと接続され、そのRTコンボーネントにテーダを送出します。 ポートル・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・						
· Detail		ポート名は、同一のコンポーネント内で、InPort, OutPort, サービスポートそれぞれの						
このセクションではケージホート申の現象を説明するトキュメントを記述します。 上のテータボートを選択すると、それぞれのドキュメントが記述できます。	ホートもに対して一度であ中必要があります。 ASCII文字が使用できます。							
ポート名:[in (InPort)	データ型: ゲータボート聞でやり取りされるデータ型を推定します。							
	Webt2006001生産使すびには、サルンEEL07~7型の同一である必要であります。 ダーダ型はOpenRTMが提供する要求型のほかに、IDLを定義することでユーザ定義型を							
*포-카필 RTC::TimedString		使用することができます。						
双数卷		(本町市・フーフルーで)、原連行(156) 年末町市を保えします。 変数の名称は言語により異なります。						
表示位置 LIFT		ポートの場所: RTSystem&ditorなどでコンポーネントを表示する際の位置を推定します。						
Documentation		「ドキュメント: データボートに関する後期を文書として記述します。						
	金でを記述する必要はありませんが、使用する人がコードを見なくても使用できる							
板寨批明:		provide the solution of the so						
2000 C								
F-9월:								
<i>9~9</i> 数:								
AT.	8							
	30							
#0:								
	10							
角生類堂, 周期:								
6.042 Ed.								
annander 11		- L						
		and and						
		mat D						
22 in								
22 in								
3								
2-	Narie	newbochy						
2) e	Narrie	diwhody						
2)e	Name	rlawischy						
	** Tamifordation 22     データスペート     Outwork(1977)     Outw	Image: Second Secon						







- 実装する言語,動作環境に関する情報を設定
- Python を選択

10(D 100(D 707-HB) (010(A) 70917H	図 第行回 ウインドウ(W) AJ	7B				
- 0 0 2 <b>3 6</b> • 4 • 1 • 1					🕄 🖏 Java 🔤 RT System Editor	ARTC Build
////-9-12270-9- 12 🖻 📚 🍸 🗂 🗆	> "NameToVelocity 23					-
NameToVelocity     RTCsml	言語・環境					
	- 23				* 121	
	COt95a2では使用する第 〇 C++ 〇 Java 〇 Patron 〇 Ruty	<b>福モ国政します</b>	T Use old build	事業: おしが4-20メモルオで変更非常増によりいた小か支援から進行可能です。 環境: 営業(との行うがの時間%)・検索ならなどの効率を見たます。 戸田情報で設定した内容(GOS編集)-6759/含額など)は、7022からAmECDA保存されます。		
	- 84					
	このセクションでは依存するう	175リや使用するOSなどを指定します		-		
	Version	05		Add		
				Delete		
	21-12-18-18-92					
	OS Version	Add CPU		Add		
		Delete		Delete		
	その他のSim #					
	CONSCIONARE			Ψ.		
	ライブラリ情報					
	Name	Version Info.		Add		
				Delete		
	基本 アクティビティ データポート	サービスボート コンフィギュレーション ドキュメント	生成 當語-環境 R3C.xx	rri		
	BuildView 11					10
					and D	
		Σlin				
		Σin				
		ΣIm				
		Σlin		Nam	rlovelocity	
		Σin		Nam	e for Adrocety	
		Дıн		Nam	n fordelocety	





### 【実習3-A】 スケルトンコードの生成

- 基本タブからコード生成ボタンを押す ことでスケルトンコードが生成される
  - Workspace¥NameToVelocity以 下に生成
    - Pythonソースファイル(.py)

       このソースコードにロボットを操作する処理を記述する
    - rtc.conf、NameToVelocity.conf
    - 以下略
  - ファイルが生成できているかを確認してく
     ださい

	≽ *Flip 🐹		
	コンポーネント型:	STATIC V	ŦŸ
	アクティビティ型:	PERIODIC v	
	コンポーネント種類:	☑ DataFlow □ FSM □ MultiMode	עב
	最大インスタンス数:	1	75
	実行型:	PeriodicExecutionContext v	19
	実行周期:	1000.0	
	概要:	↓	עב
	RTC Type :		最大
	▼ コード生成とパック	——河化	実行
	コードの生成およびパ コード生成 パッケ	ッケージ化を行います。 ージ化	概要 RT(
	▼ プラマアイル情報の	)インポート・エクスポート	
-1	ド生成」ボタン	<u>ノをクリックする</u>	コー パッ
	く スタンティドティ デー	タポート サービスポート コンフィギュレーション ドキュメント牛成 言語・環境 RTC	.xml
_			





変数初期化 (OpenRTM-aist 1.2.0では 自動生成される予定です)



 NameToVelocity.py をエディタで開き, 変数初期化部分を下記のように修正します

def \_\_init\_\_(self, manager):
 OpenRTM\_aist.DataFlowComponentBase.\_\_init\_\_(self, manager)

self.\_d\_in = RTC.TimedString(RTC.Time(0,0), "")
self.\_inIn = OpenRTM\_aist.InPort("in", self.\_d\_in)

self.\_d\_out = RTC.TimedVelocity2D(RTC.Time(0,0), RTC.Velocity2D(0.0, 0.0, 0.0))
self.\_outOut = OpenRTM\_aist.OutPort("out", self.\_d\_out)







 NameToVelocity.py をエディタで開き, Activate, Deactivate 部分を下記のように修正します
 – 起動,終了時にロボットを停止するための処理



def onDeactivated(self, ec\_id): #ロボットを停止する self.\_d\_out.data.vy = 0.0 self.\_d\_out.data.vx = 0.0 self.\_d\_out.data.va = 0.0 self.\_outOut.write()

return RTC.RTC\_OK





### 【実習3-A】 ソースコードの修正③

- Execute部分を 右記のように修正 します
  - 認識した文字列
     に制御速度を割
     り当てます
  - 割り当てる速度 や文字列は自由 に変更してみてく ださい

def onExecute(self, ec\_id): # 入力データが存在するか確認 if self. inln.isNew(): # 入力データが存在する場合には、データを別変数に格納 data = self.\_inln.read() #入力データの文字列に応じてロボットを操作する if data.data == "bow tie": self. d out.data.vx = 0.5elif data.data == "hook": self. d out.data.vx = -0.5elif data.data == "pinwheel": self. d out.data.va = 0.3elif data.data == "envelope": self. d out.data.va = -0.3else: self. d out.data.vx = 0 $self._d_out.data.vy = 0$ self. d out.data.va = 0認識した文字列に対して 速度を割り当て self. outOut.write() return RTC.RTC\_OK





### 【実習3-A】 コンポーネントの起動

コンポーネント名	起動方法
NameToVelocity	作成したNameToVelocity.pyファイルをダブルクリックもしくはコマンドプロ ンプトから"python NameToVelocity.py"と入力して起動して下さい.
OpenCVCamera	OpenRTM-aistインストール時に同時にインストールされています. Windowsの検索(Windowsアイコン押下時のプログラムとファイルの検 索など)を用い, OpenCVCameraComp.exeを起動して下さい.
CameraViewer	OpenRTM-aistインストール時に同時にインストールされています. Windowsの検索(Windowsアイコン押下時のプログラムとファイルの検 索など)を用い, CameraViewerComp.exeを起動して下さい.
ImageToObjectP rediction	USBメモリで配布されたsampleフォルダにある ImageToObjectPrediction.pyをダブルクリックして起動して下さい.
RaspberryPiMou seSimulator	USBメモリで配布されたEXEフォルダにある RaspberryPiMouseSimulatorComp.exeをダブルクリックして起動し て下さい.





### 【実習3-A】データポートの接続

- RTSystemEditor を起動し, 右記のようにポートを 接続します
- 接続後、コンポーネ ントをActivateしま す









- 起動したコンポーネントが全てActive状態(緑色)
- カメラ画像が表示される
- カメラ画像が認識された際にImageToObjectPredictionコン ポーネントを起動したプロンプト上に"Recognized Object: xxx" と表示される
- コード上に割り当てた物体を認識すると、シミュレータ上のロボットが 期待通り動作する





実機での動作確認

- ※時間がある方は,下記にも挑戦ください ・身の回りのより認識しやすい物体に置き換えて動作
- ・制御方法(条件分岐)をより複雑化してなめらかな動作









# 4. 実習B 「データ収集・蓄積の検証」





### 【実習3-B】「データ収集・蓄積」実習







## 【実習3-B】「データ収集・蓄積」実習

L D T = U Une y-	7B-	101 11214	LR. (Osluchia	imaç	e_20171124					08	50	à				senadouse	Excel				.7	HE -	a x
	1) + 7-tf- + ta	ishashi SELUT	パターを作成します。	image 20	71174			v 6	imade 2 0	77410 1		ページレイアリト	数式 データ	校開	表示 - アF4	> ACROE	AT #⇒∆						サインイン
<ul> <li>☆ お気に入り</li> <li>※ おうしょ</li> <li>※</li></ul>	0.prg	1.pog	2.png	3.png	4.pog	5.png	6.png	7.prg	Range A		MS P33	** 1 ⊻·⊞	• <u>n</u> • <u>A</u> •	А <u>х</u> -		= ₽ = ₽ *	<b>63</b>		記 Series 1117-73 11111-73 11111-73 11111-73 11111-73 11111-73 11111-73 11111-73 11111-73 11111-73 111111-73 1111111111	き書式。 として書式設定。 いたん。	部 #3入 ・ (新) #1版 ・ (新) 書式・	Σ· 17. 	3
CheDrive (Mac)		1200.9	120.9	1200	1.0.4	1.0.	1.0.4	1.0.4	1.0.4	At	+] : [X	P fx 11			10.00		-97.65	. 14		2110		- PROC	*
▲ OneDrive - 国立研究開発法人產業技術総合			1 miles	IL.S		The second	and the	100-100	and and		8	C D		P	G	H 1	1	ĸ	- 10	м	N O	P	Q (4)
Ke FC     Decktop     Decumets     Documets     Movies     Music     Returnet     Gr-hr/#7470 (C:)     Chain #7470 (C:)     Chain #62(Ki)     MobieBockaps on Mac' (V:)     Chain Mac' (X:)     Alteries on Mac' (Z:)     Alteries on Mac' (Z:)	9.png 18.png 27.png 27.png 36.png	10.png 19.png 28.png 28.png 23.png 37.png	11.png 20.png 20.png 20.png 20.png 20.png	12.prg 21.prg 20.prg 30.prg 39.prg	13.prg 22.prg 23.prg 23.prg 40.prg	14.prg 23.prg 32.prg 32.prg 41.prg	15.prg 24.prg 33.prg 42.prg	16.prg	17.grg 26.prg 35.prg 41.prg	1 1 2 3 4 4 4 1 5 5 1 6 6 1 7 3 8 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 -1978 1 -	-1978 -1978 -1978 -1978 -1976 -1976 -1976 -1976 -1976 -1976 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978	11 11 11 11 11 11 11 11 11 11										
	45.png 54.png	46.png 46.png 55.png	47.pog	48.png 48.png 57.png	49.prg	S0.prg	51.prg	52,prg	52.prg	20 20 21 22 23 23 24 25 25 25 25 27 26 29 20	1 -1978 1 -1970 1 -1970 1 -1978 1 -	-1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1970 -1970 -1970 -1970 -1970 -1970	11 11 11 11 11 11 11 11 11 11										
	63,png 63,png 72,png	61.png	65.png 65.png 74.png	66,png 66,png 75,png	67,png	68.png	69.prg	70.png	71.prg	31 1 32 33 1 34 1 36 1 37 1 38 1 39 1 40 1	1 -1978 1 -1978	-1978 -1978 -1976 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978											
	81.png 90.png	82.png 61.png	83.png	84.png	85.prg	Bó.prg Bó.prg S5.prg	87.prg	58.prg	89.prg	43 44 145 146 147 148 147 148 147 148 149 149 149 150 149 149 149 150 149 149 149 150 149 149 150 149 149 150 149 149 150 149 149 150 149 149 150 149 149 150 149 149 150 149 149 150 149 149 150 149 149 149 149 150 149 149 149 149 149 149 149 149 149 149	1 -1978 1 -1978	-1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978	11 11 11 11 11 11 11 11 11 11 11										
	99.png	100.png	101.png 110.png	102.png	103.png 112.png	104.png 104.png 113.png	105.png 105.png 114.png	106.prg	107.png IOT.png III6.png	51           52           53           54           55           56           57           58           50           60	1 -1978 1 -1978	-1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978 -1978	11 11 11 11 11 11 11 11 11 11										
121 億の靖日	H	HELCE	H CE	1 ALCEN					=	3457	sensor							11 11			8 🛄	- <b>1</b>	+ 100%

NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)





#### 【実習3-B】 修正するRTC(ImageDataCollector)の仕様

- カメラ画像(CameraImage型)が入力され、入力されたカメラ画 像をPNGファイルとして保存します
- ロボットの赤外線センサ情報(TimedShortSeq型)が入力され、 入力されたセンサ情報をCSVファイルとして保存します
- 上記処理を100ms毎に行います









- RTC Builderによるソースコード等のひな型の作成
   一既存のプロファイルを再利用
- ソースコードの編集
  - ImageDataCollector.pyの作成
- RTシステムエディタによるRTシステム作成、動作確認
   RTシステム作成
   データポート接続
- Raspberry Piマウスシミュレータとの接続、動作確認
- Raspberry Piマウス実機との接続、動作確認







 ImageDataCollectorコンポーネントのスケルト ンコードをRTCBuilderで作成する







### 【実習3-B】 基本プロファイルの入力

- USBメモリ sampleフォルダ内のImageDataCollector
   コンポーネントのRTC.xmlを読み込む
  - RTC Builder には既存コンポーネントのプロファイルを読込む機能があります
  - 「基本」タブにある「プロファイル情報のインポート・エクスポート」の「インポート」 ボタンをクリックし,改造元のImageDataCollectorにあるRTC.xmlを選択

• B B L 2 4 4 • 8 • 8 • 8 • 8	0.0				0497-7722 환 화ve ARC Build			
Nat-1-19230-3-11 E S 7 5 0	> ImageSensorDa	taColactor > ImageDataColactor 12			and the second			
Co ImageDetaColector	**							
<ul> <li>Criske</li> <li>Criske modular</li> </ul>	a fil-famour	of Basic Boofile		12 I.S.A.				
Es cpack_resources	TOMAL OF THE	and an american set of the		Trank .	and assembly identified by			
e 🔄 doc	*Eller St.	Imaginate Colorer		Exa. MG +	この名称はコンポーキントのペースインスタンス名にも使用されます。			
> Co- indude	School Street	Calasta sontas data fanta campas imaso		TT: 188.	使用できる文字はアルファベット、数字、ハイフン、アンダース3アのみです。			
> 😂 STC	*17. Tar. 1	100		Con magn	ASCII文字与使用できます。			
COPYING		Internation		N-902 .	RTコンボーキントのパーションを規定します。 メンゴンション教育のの形式でき、アレア(FNI)。			
COPYING LESSER	TECONO DE DE CAL	Columber		1398:	AT122ポーキンドを作用した作用名、ベンダ名を無定します。			
ImageDataCollector - 20 - py		(47-940			ASCII文字U使用Cdatg.			
ImageDataCollector.py	200-2/11:	SIAIL.	÷		第代時にない場合は任意のパテリンを見入力することができます。			
ImageOstaCollector.py20171122111333	7974C741	Polaceic	*	Constanting of the	★月できな文字は、アルファウォン、数字、パイプン、アンタースコアのわてす。 なてついた。まいたのであります。			
ImageOstaColector.py20171122135105	コンポーネント構成さ	CataFlow PSM MultiMode		-	・STATIC 単的に生成用目されに、RTC ・LPEQUE 動的に生成用目を行きユニークSRTC			
README.ImageDateCollector	最大インスタン2数:	1 [1		7774748	-この時間のはいいに、動きに工業の変化化で RT121ポーキントのフラナバナイロネルデします。			
READINE Integrate control 201711221398	#179 I	PeriodidExecutionContext		1000000000000	FERIDOIC: 第行電域外面含の環境電行電流TC			
READVE.ImageDataCollector201711221754	ARTER :	1000.0			-EVENTERIVEN: 475-HEIMERTC			
rtc.conf	1.000		1	コンボーキント種類:	<ul> <li>れてコンボーネンドの優勝を指定します。</li> <li>Data DowComponent: 常知事が形ちて</li> </ul>			
A RTC.aml	81.M 1				-FiniteStateMachine 新聞状態連時如下C			
RTC.arv/20171122134922	BTC Tune I			#たら2028:	- PRUCINOSE: マルチモード当然でし、 - を成月前台インスタンス数を始定します、 新聞がない場合はらを用定します。			
RTC.3r9201712213536				展行型:	Riffethelut,			
	· J-71002/09			<b>英行讯期</b> :	コンポーキントクウションの実行用用を行うの目的にであっています。			
		W7-Siberty at a		40.00 i	RT12ンポーマントの標準な説明を記述します。			
				RTC Type :	物定機能を実現するKTCの機能を記形する必要がある場合に使用します。			
	* 30374 LONG	0438-1-1028-1			Example Contraction of the second sec			
	7027416月時04	ンボート名は以上のスポートを行います。		コード生成: 設定した情報を基にRTCのスケルトンコード生生成します。				
	128-1 102	<i>A</i> -1		パックージ化: RTCOソースコード、実行用パイナリコードなどを基準形式でアーカイプします。				
	BA 797/67/ 7-	- 9月-1 サービスポート コンスドネシーション ドキスメントを相 業務・数	R RTC.xml					
	Buildview 32				-			
		3 mm						
		22 mass						
			Inegri	DeteCollector				





### 【実習3-B】データポートの設定

- 以下のOutPortを追加する
  - ポート名: sensor
  - データ型: RTC::TimedShortSeq
  - 他の項目は任意
- 以下のOutPortはそのまま維持
  - ポート名:image
  - データ: RTC::CameraImage
  - 他の項目は任意

0= 2	• 10 ( p. • 10) •			ウイック・アクセス (1) AT RT 5	System Editor 🏠 RTC Builder 🔲 RTC Debugger
Control C			● イナンフロビー         ● イナン クレビー         ● イナン クレー         ● イナン クレー	iyotem Editor () RTC Bulder () RTC Delvoger () RTC Delvoger () RTC Delvoger () RTC Delvoger () () () () () () () () () () () () ()	
	デー検索:     思味:     認味:     認味:     現成:     現成:     現成:     現成:     現成:     現成:     現成:     現成:     現成:     電流:     フランパフィーマン・プラン・プラーン     登 加点ののか □ プンパフィーマン・プングラーン	(チュレーション) ドキュバント意成   重勝・磁策	RTC.aml		
	BingeCotaColector     Boytecrofler     Boytecrofler     Boytecrofler     Boytecrofler     Boytecrofler     Boytecrofler				





### 【実習3-B】 スケルトンコードの生成

## 実行周期が10になっていることを確認 10Hz=100msごとに動作

- 基本タブからコード生成ボタンを押す ことでスケルトンコードが生成される
  - Workspace¥ImageDataCollector
     以下に生成
    - Pythonソースファイル(.py)
       このソースコードにロボットを操作する処理を記述する
    - rtc.conf、ImageDataCollector.conf
    - 以下略
  - ファイルが生成できているかを確認

	≽ *Flip 🐹		
	コンポーネント型:	STATIC ¥	モジュ
	アクティビティ型:	PERIODIC V	
	コンポーネント種類:	☑ DataFlow □ FSM □ MultiMode	コンポ
	最大インスタンス数:	1	アクテ
	実行型:	PeriodicExecutionContext V	
	実行周期:	10	
	概要:		コンポ
	RTC Type :		最大
	▼ コード生成とパック		実行
	コードの生成およびパ	ッケージ化を行います。	美行
	コード生成 パッケ		概要
		、 ハーゴ L ートフーゴ L	RTC
	▼ 1711/1月報0	J1 2M-F-19XM-F	
-1	ド生成」ボタン	ンをクリックする <sup>)。</sup>	コード パッケ
			×.4
	< 基本 アクティビティ デー	タポート サービスポート コンフィギュレーション ドキュメント生成 言語・環境 RTC	.xml
į	基本」タブを遭	選択	







 ImageDataCollector.py をエディタで開き, 変数初期化部分を下記のように修正します









• ImageDataCollector.py をエディタで開き, 下記のように修正します







【実習3-B】 ソースコードの修正③

● ImageDataCollector.py をエディタで開き,下記のように修正します







### 【実習3-B】 コンポーネントの起動

コンポーネント名	起動方法
ImageDataCollec tor	作成したImageDataCollector.pyファイルをダブルクリックもしくはコマンド プロンプトから"python ImageDataCollector.py"と入力して起動して 下さい.
OpenCVCamera	OpenRTM-aistインストール時に同時にインストールされています. Windowsの検索(Windowsアイコン押下時のプログラムとファイルの検 索など)を用い, OpenCVCameraComp.exeを起動して下さい.
CameraViewer	OpenRTM-aistインストール時に同時にインストールされています. Windowsの検索(Windowsアイコン押下時のプログラムとファイルの検 索など)を用い, CameraViewerComp.exeを起動して下さい.
RobotController	第二部で作成したコンポーネントです. RobotController.pyをダブルクリックして起動して下さい.
RaspberryPiMou seSimulator	USBメモリで配布されたEXEフォルダにある RaspberryPiMouseSimulatorComp.exeをダブルクリックして起動し て下さい.





【実習3-B】 データポートの接続

RTSystemEditorを起動し、下記のようにポートを接続します
接続後、コンポーネントをActivateします









- 起動したコンポーネントが全てActive状態(緑色)
- カメラ画像が表示される
- RobotController のスライドバーでロボットが制御できる
- ImageDataCollectorを起動したフォルダ以下にカメラ画像ファイルが生成され,時間経過とともにファイル数が増える
- ImageDataCollectorを起動したフォルダ以下にセンサデータファ イルが生成され,時間経過とともにファイルサイズが増える

全てOK



実機での動作確認

- ※時間がある方は、下記にも挑戦ください
- ・OpenCVCameraのconfigurationを変更し カメラ画像のサイズを大きく or 小さくしてみる









# 5. 応用課題

PAIST



### 【応用課題①】 RaspberryPiMouseのカメラを使う

- RaspberryPiMouse に USB カメラを接続し、SSH でログインして、OpenCVCameraコンポーネントを起 動します
- 実習A or B と同じようにコンポーネントを接続して動作 させます







### 【応用課題②】認識成功時の画像を保存する

 ImageDataCollector と ImageToObjectPredictionのコードをマージして, 認識に成功した場合の画像を保存するコンポーネントを 作成する

