



単体ロボット向けプロトタイプの 導入手順と使用方法の紹介

2020/07/18 辻 悠斗 (永和システムマネジメント) 森 崇 (永和システムマネジメント)

© Copyright 2020, ESM, Inc.



箱庭

はじめに

単体ロボット(ETロボコン)向けシミュレータの構成

マイコン・	シミュレ	ータ
-------	------	----



EV3RT

ASP3/ASP

athrill



ETロボコンを題材として構築

技術研鑽視点での狙い:

- ・物理シミュレータとマイコンシミュレータ間の 連携方法の検討
- ・異なるシミュレータ間の時間同期の検討 その他の狙い:
 - ・ETロボコンユーザ層に箱庭を広める(広報活動)

© Copyright 2020, ESM, Inc.

Unityパッケージの設計と作成にあたっては, 宝塚大学東京メディア芸術学部 吉岡章夫准教授 および学部生の杉崎涼志さん,木村明美さん, 千葉純平さんにご協力いただきました。

HackEVのUnityアセットは、ETロボコン実行委員会 より提供いただいたデータを基に作成しています. 実行委員会の皆さまに深く感謝いたします. ただし本アセットはETロボコンの本番環境とは異なりま すので、大会に参加予定の方はご注意ください. また、本アセットは、個人利用または教育利用に限 定してご利用ください.





はじめに

単体ロボット(ETロボコン)向けシミュレータの構成

マイコン・シミュレータ	× Persp
制御処理(C/C++)	8
EV3RT	
ASP3/ASP	
athrill	unity
FTロボコンを題材として構築。	Unityパッケージの設計と作成にあたっては,

今からこのプロトタイプの 環境構築手順を紹介します Unityパッケージの設計と作成にあたっては, 宝塚大学 東京メディア芸術学部 吉岡章夫准教授 および学部生の杉﨑涼志さん,木村明美さん, 千葉純平さんにご協力いただきました。

HackEVのUnityアセットは、ETロボコン実行委員会 より提供いただいたデータを基に作成しています. 実行委員会の皆さまに深く感謝いたします. ただし本アセットはETロボコンの本番環境とは異なりま すので、大会に参加予定の方はご注意ください. また、本アセットは、個人利用または教育利用に限 定してご利用ください.





はじめに

- 紹介するプロトタイプの構成 使用プラットフォーム
 - Windows(WSL), Linux, Mac
 - ターゲットCPU
 - V850, ARM
 - Unityとの通信方式
 - UDP, MMAP

今回は以下の構成での 環境構築手順を紹介します ・Windows(WSL) ・V850 ・MMAP

MMAPはUDPよりもシミュレー ション精度が高いという理由で, 今回は採用しました

- 導入手順でのつまづきポイントも紹介します
 - •Webサイト上ではお伝えできていない細かい注意点 についても今回お伝えできたら良いと思っています



TOPPERS
Toyohashi OPen Platform for Embedded Real-time Systems

アジェンダ



1. 単体ロボット向けシミュレータの導入手順

こちらに記載の導入手順を紹介します

https://toppers.github.io/hakoniwa/single-robot-setup/singlerobot-setup-index/

2. 単体ロボット向けシミュレータの使用方法・デモ

基本的にはこちらに記載の使用方法を紹介します

https://toppers.github.io/hakoniwa/single-robotusage/single-robot-usage-index/





単体ロボット向けシミュレータの導入手順

Windows V850を使用する場合の手順です





- 1. WSLのインストール
- 2. Rubyのインストール
- 3. athrill2のインストール
- 4. V850用クロスコンパイラのインストール
- 5. 箱庭用EV3RT環境のダウンロード
- 6. Unityのインストール・初期設定



1.WSLのインストール





• WSL1をインストールします

Ubuntu 18.04 LTSをMicrosoft Storeからインストールします

- ・インストール直後に、sudo apt updateを行っておきます
 - ・後ほどgccやmakeをインストールするのに必要となります



2.Rubyのインストール



※今回紹介するプロトタイプはカーネルがASP3ですので, Rubyの使用が前提となります

\$ sudo apt install ruby



3. athrill2のインストール



- athrillはターゲットCPUに依存する実装とそうでない 実装で分けられています
- 1. athrill(ターゲット非依存部)のチェックアウト \$ git clone https://github.com/toppers/athrill.git
- 2. athrill(ターゲット依存部)のチェックアウト

\$ git clone https://github.com/toppers/athrill-targetv850e2m.git



3. athrill2のインストール



- athrillをビルドするために,gccとmakeをインストールします
 \$ sudo apt install gcc
 \$ sudo apt install make
- athrillをビルドします
 \$ make timer32=true clean
 \$ make timer32=true

 ・ 必ず、timer32オプションを つけること! つけないと制御アプリが上手く 動作しません

athrillを使用するために環境変数を設定します

例:export PATH=<athrill配置フォルダパス>/athrill/bin/linux:\${PATH} .bashrcなどに登録しておきます



4.V850用クロスコンパイラのインストール





Latest release	athrill-gcc(linux64bit v850-gcc/libc.a is original)	
♥ v1.1 - 0- 57d10b4	Stmori released this on 14 May · 1 commit to master since this release	
Verified	libc.a reverted to original one.	
Compare 🔻		
	- Assets 3	
		225 MB
	Source code (zip)	
	Source code (tar.gz)	

https://github.com/toppers/athrill-gccv850e2m/releases/tag/v1.1



4.V850用クロスコンパイラのインストール



- ・ダウンロードしたコンパイラを解凍し、コピーします
 - ・一度解凍したパッケージ内にさらに圧縮ファイルがあります
- \$ tar xzvf athrill-gcc-package.tar.gz
- \$ cd athrill-gcc-package/
- \$ tar xzvf athrill-gcc.tar.gz
- \$ sudo mv usr/local/athrill-gcc /usr/local







cloneする場所は,以下のように athrillと同じフォルダ階 層で実施してください.

---athril

---athrill-target-v850e2m

L---ev3rt-athrill-v850e2m

\$ git clone https://github.com/toppers/ev3rtathrill-v850e2m.git

© Copyright 2020, ESM, Inc.



5. 箱庭用EV3RT環境のダウンロード



 今回使用するEV3制御アプリをダウンロードし、箱庭EV3RT環境 にコピーします

サンプルアプリのclone

\$ git clone https://github.com/toppers/hakoniwa-scenariosamples.git

コピー元

hakoniwa-scenario-samples/single-robot 配下の全てのフォルダ

コピー先

ev3rt-athrill-v850e2m/sdk/workspace 配下





UnityのバージョンはUnity 2020.1.0b9 (64-bit)以降 をインストールしてください





- アセットのダウンロードをします
- single-robot-HackEV.unitypackage
 - <u>https://github.com/toppers/hakoniwa-Unity-</u> <u>HackEV/releases</u>

- ev3rt-simple-robot.unitypackage
 - <u>https://github.com/toppers/hakoniwa-Unity-</u> <u>SimpleCar/releases</u>









Unityを起動し、新規プロジェクトを作成します

 Unity Hub 				- [×
🚭 unity				\$	Θ
⑦ プロジェクト	プロジェクト		リストに追	加新規作成	-
 ◆ 使い方を学ぶ ** つミューニック 	プロジェクト名	Unity バージョン	ターゲット	最終更新 个	Q
= インストール	check_for_v1.0 C:\Users\tmori\check_for_v1.0 Unity バージョン: 2019.3.4f1	2019.3.4f1 👻	使用中のプラッ 👻	a few seconds ago	:
	check_robocon2 C:\Users\tmori\check_robocon2 Unity バージョン: 2019.3.4f1	2019.3.4f1 👻	使用中のプラッ… 👻	9 minutes ago	:
	check_robocon C:\Users\tmori\check_robocon Unity バージョン: 2019.3.4f1	2019.3.4f1 👻	使用中のプラッ… 👻	43 minutes ago	:
	ETRoboconSimulator C:\Users\tmori\ETRoboconSimulator Unity バージョン: 2019.3.4f1	2019.3.4f1 👻	使用中のブラッ… 👻	5 days ago	:
	test_latest_version				





パッケージのインポート

 Unity のメニューから、「Assets」⇒「Import Package」⇒
 「Custom Package…」と選択し、任意の unitypackageファ イルをインポートします







Project/Scenes配下にToppers_Courseというシーンが追加されます







シミュレーションに関わる設定をします

• Unity のメニューから, 「Edit」⇒「Project Settings」を選択し ます

Time Fixed Timestep を 0.001に, Time Scale を 0.6に設定します

O Project Settings			-= 🗖 ×
		(Q)
Audio	Time		💽 7 ¢,
Editor			
Graphics	Fixed Timestep	0.001	
Input	Maximum Allowed Timestep	0.3333333	
Physics	Time Scale	0.6	
Physics 2D	Maximum Particle Timestep	0.03	
Player			
Preset Manager			
Quality			
Script Execution Order			
Tags and Layers			
TextMesh Pro			
Time			
VFX			





シミュレーションに関わる設定をします

• Unity のメニューから, 「Edit」⇒「Project Settings」を選択し ます

Quality OtherのVSync Count を Don't Sync に設定します

Preset Manager Quality Script Execution Order	Texture Streaming Shadows			
Tags and Layers TextMesh Pro Time VFX	Shadowmask Mode Shadows Shadow Resolution Shadow Projection Shadow Distance	Distance Shadowmask + Hard and Soft Shadows + High Resolution + Stable Fit + 150		
	Shadow Near Plane Offset Shadow Cascades Cascade splits	3 Four Cascades +		
	Other Skin Weights VSync Count LOD Bias	4 Bones		

© Copyright 2020, ESM, Inc.





シミュレーションの通信方式(MMAP)に関わる設定をします • Unity のメニューから,「Edit」⇒「Project Settings」を選択します

Player Other SettingのScripting Define Symbols に"VDEV_IO_MMAP" と設定します

O Project Settings			-= 🗖 🗙
		(Q	
Audio Editor		Mono	💽 🕂 🌣
Graphics	Api Compatibility Level*	.NET Standard 2.0	•
Input Physics Physics 2D	C++ Compiler Configuration Use incremental GC (Experimental) Disable HW Statistics*	Release	*
Player Preset Manager Quality	Scripting Define Symbols VDEV_IO_MMAP		
Script Execution Order Tags and Layers	Allow 'unsafe' Code Active Input Handling*	Input Manager	
TextMesh Pro Time	Optimization	-	

入力したあとは, カーソルを外すなどしてください. うまく設定が反映されない場合があります

© Copyright 2020, ESM, Inc.





- シミュレーションの通信方式(MMAP)に関わる設定をします
- Unity のHierarchyビューにてRoboModel_3を選択します
- ・選択すると、画面右のInspectorビューに[EV3 Sensor (Script)]
 と[EV3 Actuator (Script)]が表示されます







シミュレーションの通信方式(MMAP)に関わる設定をします

 ・画面右のInspectorビューにて[EV3 Sensor (Script)]と[EV3
 Actuator (Script)]のFilePathにmmapファイルの絶対パスを入力
 します

Line_trace X		
名前	^	更新日時
app.c		2020/06/1
📄 app.cfg		2020/06/1
📄 app.h		2020/06/1
📄 athrill_mmap.bin		2020/06/1
device_config.txt		2020/06/1
📄 device_config_mn	nap.txt	2020/06/1
📄 log.txt		2020/06/2
Makefile.inc		2020/06/1
memory.txt		2020/06/1
memory_mmap.tx	t	2020/06/1
unity_mmap.bin		2020/06/1

🔻 # 🗹 EV3 Sensor (Script)				:			
Script	■ EV3Sensor			۲			
Filepath	rkspace\line_trace\unity_mmap.bin						
🔻 # 🗹 EV3 Actuator (Script)				:			
Script	■ EV3Actuator	■ EV3Actuator					
Filepath	rkspace\line_trace\athrill_mmap.bin						

EV3 Sensorにはunity_mmap.binを, EV3 Actuatorにはathrill_mmap.bin の絶対パスを入力します





単体ロボット向けシミュレータの使用方法

Windows V850を使用する場合の手順です

- 1. EV3制御プログラムのビルド
- 2. Unityのシミュレータの起動
- 3. athrillの起動





1.EV3制御プログラムのビルド

- WSL上でev3rt-athrill-v850e2m/sdk/workspace に 移動します
- 下記コマンドでビルドします
 \$ make img= <アプリケーションフォルダ名> clean
 \$ make img= <アプリケーションフォルダ名>





2.Unityのシミュレータの起動

・Unityの画面上で、実行ボタンを押下します

ckev-demo - Toppers_Course - PC, Mac & Linux Standalone - Unity 2020.1.0b11.3880 Personal [PREVIEW PACKAGES IN USE] <DX11>

dit Assets GameObject Compo	onent	Window He	р									
🕂 🖸 🖾 🗯 🗙		Center 🔀 L	ocal 🛙 🏥	5					II	M		
rarchy	3 :	# Scene	📾 Gam	e	📄 Asse	t Store						
ि अर All		Shaded	•	2D	• I•	\$ ▼	ø 🕫				×	
 Toppers_Course Main Camera Directional Light Reflection Probe Light Probe Group Directional Light (Sub) Robot Course Red Blue Green 	:								*			





2.Unityのシミュレータの起動

・ 下図のように, 画面が切り替わります







3. athrillの起動

• WSL上でathrillを起動し、シミュレーションを開始します

ev3rt-athrill-v850e2m/sdk/workspace/<アプリケーションフォルダンに移動して、以下のコマンドを実行します

【通信方式がMMAPの場合】

\$ athrill2 -c1 -t -1 -m memory_mmap.txt -d
device_config_mmap.txt ../asp

たった1コマンド叩くだけでシミュレーションが実行できる!





3. athrillの起動

・成功するとWSL上で以下のメッセージが出力され、Unity上の画面で、EV3のロボットが動き始めます

core id num=1
ROM : START=0x0 SIZE=512
RAM : START=0x5ff7000 SIZE=10240
Elf loading was succeeded:0x0 - 0xfd68 : 63.360 KB
ELF SYMBOL SECTION LOADED:index=22
ELF SYMBOL SECTION LOADED:sym_num=964
ELF STRING TABLE SECTION LOADED:index=23
athrill device func call=0x60f7444

TOPPERS/ASP3 Kernel Release 3.2.0 for V850-ESFK3 (Nov 6 2019, 10:56:14) Copyright (C) 2000-2003 by Embedded and Real-Time Systems Laboratory Toyohashi Univ. of Technology, JAPAN Copyright (C) 2004-2017 by Embedded and Real-Time Systems Laboratory Graduate School of Information Science, Nagoya Univ., JAPAN

brick_dri initialized.

/ __/ | / /_ // _ \/_ __/ / _/ | |/ //_ </ , _/ / / /__/ |___/___/_/|_ / _/

Powered by TOPPERS/HRP2 RTOS Initialization is completed.. System logging task is started.





[条件] Unityのシミュレーション精度: 5msec 制御アプリケーション周期 :10msec

HackEVのUnityアセットは、ETロボコン実行委員会より提供いただいたデータを基に作成しています. 実行委員会の皆さまに深く感謝いたします. ただし本アセットはETロボコンの本番環境とは異なりますので、大会に参加予定の方はご注意ください. また、本アセットは、個人利用または教育利用に限定してご利用ください.





謝辞·特記事項

- Unityパッケージの設計と作成にあたっては、宝塚大学東京メディア芸術学部 吉岡章夫准教授および学部生の杉﨑涼志さん、木村明美さん、千葉純平さん にご協力いただきました。
- HackEVのUnityアセットは、ETロボコン実行委員会より提供いただいたデータを 基に作成しています。実行委員会の皆さまに深く感謝いたします。
 ただし本アセットはETロボコンの本番環境とは異なりますので、大会に参加予定の 方はご注意ください。また、本アセットは、個人利用または教育利用に限定して ご利用ください。
- 本資料は、ユニティテクノロジーズまたはその関連会社がスポンサーとなったり、
 ユニティテクノロジーズまたはその関連会社と提携しているものではありません。
 本資料に掲載された Unity の登録商標一覧 に含まれる Unity の登録商標は
 すべて、ユニティテクノロジーズまたはその米国や他の国々に所在する関連会社の
 登録商標または商標です。