Sensor Fusion with RTMaps on an Embedded Platform

Jacob Perrin and Theo King
Applications Engineers

dSPACE Inc. · 50131 Pontiac Trail · Wixom, MI · 48393 USA
1. RTMaps Embedded Overview
2. Demo: RTMaps on BlueBox
3. RTMaps Application Processing
1) To tag recorded data means to mark data files with dedicated attributes like highway, roundabout, crossing, environmental conditions etc. so that you can easily search for these attributes and files in large databases.

Recorded data
(Sensor, vehicle network, reference data) + time stamps, tags, ...
Terminology

- **Component**
  - Smallest atomic execution unit
  - Typically executed in a separate CPU thread

- **Package**
  - Container including multiple components
  - Technically a shared library (.dll / .so) extended by a RTMaps signature
### RTMaps – Software Architecture

**PC**
- RTMaps SDK
- RTMaps Components library
  - Numerous off-the-shelf components
- RTMaps Execution Engine
  - Clock management
  - Components management
- Operating System
  - Architecture: x86, x86_64
- RTMaps Studio
  - Diagnostics
  - Threads & events management
  - Buffers management
  - Performance monitoring

**Embedded target**
- RTMaps Components library
- RTMaps Execution Engine
- Operating System
  - Architecture: x86, x86_64
  - ARM

Confidential
RTMaps Remote Studio Connector

- Use RTMaps Developer Version on your PC to cross-develop algorithms and collect data directly on embedded platform
- Use RTMaps remote studio connector to remote control RTMaps on embedded platform
- Use RTMaps Run-Time Version on embedded target platform to execute RTMaps diagram disconnected from Developer PC

RTMaps Developer Version
  - Remote connect to the embedded platform

PC platform

RTMaps runtime version plus studio connector
  - Development of diagrams

Embedded x86 or ARM platform

TCP/IP

RTMaps diagram

RTMaps remote studio connector

RTMaps execution engine
Explicit clock synchronization
- One master, multiple slaves
- Clock distribution from master to slave
  - Exchange of synchronization frames (TCP/IP)
- Optional master clock synchronization on an external time source (e.g., UTC via GPS)
Data Processing on Distributed Platforms

- The distribution configuration window

RTMaps master

RTMaps slave
NXP BlueBox 2.0

**Computing:** NXP Blue Box v2.0
- LS2: Layerscape processor, Ethernet aggregation and high end ARM core complex (8 x A72 ARMv8 cores)
- S32V: Vision processor for Automotive (vision pipeline (APEX) + 4 x ARM A53 cores)
- S32R: Safety processor, ASIL-D capable

**Operating System:** Auto SDK Linux + Ubuntu RFS.
At a high-level, the BLBX2 consists of three major components integrated together to provide a powerful, yet flexible, and safe development platform for automotive and industrial applications.

- The **LS2** provides high single and multithreaded performance with eight ARM A72 cores.
- The **S32V** features a massively parallel Vision accelerator known as the APEX2, which is well suited for image processing algorithms and light machine learning inference work.
- The **S32R** is offered as an ASIL-D supervisor that complements the ASIL-B ready S32V.
DEMO: RTMAPS ON BLUEBOX
RTMaps Application Processing and Playback
Why RTMaps for application development?

RTMaps allows for cross-platform development (Windows or Linux OS / x86 or ARM processors)

RTMaps was created with the following in mind:

- Dynamism
- Speed
- Time
- C++ characteristics
What are the main benefits of RTMaps?

RTMaps is
- more efficient in terms of CPU consumption
- more intuitive to operate and easier to program in C++, Python, Simulink
- continuously being further developed according to customer requirements

RTMaps provides
- off-the-shelf component libraries for sensors and data processing, e.g., for OpenCV
- a continuous transition from prototyping on PCs to production close implementations on embedded ARM platforms
- open and documented APIs and file formats
- compatibility with ROS and EB Assist ADTF
RTMaps Components - Python

# This sample adds two images. Note that we divide by two before adding to avoid overflows.
# "/" is the integer division in Python. If you use "/", the output will be float and automatically
# converted to Matrix, and not IplImage
from rtmaps import *
# Python class that will be called from RTMaps.
class rtmaps_python:

# Birth() will be called once at diagram execution startup
def Birth(self):
    pass

# Core() is called everytime you have a new input
def Core(self):
    self.output_0 = self.input_0//2 + self.input_1//2

# Death() will be called once at diagram execution shutdown
def Death(self):
    pass
#include "maps_ComponentTemplate.h"

// Use the macros to declare the inputs
MAPS_BEGIN_INPUTS_DEFINITION(MAPSCOMPONENTTEMPLATE)
   //MAPS_INPUT("iName",MAPS::FilterInteger32,MAPS::FifoReader)
MAPS_END_INPUTS_DEFINITION

// Use the macros to declare the outputs
MAPS_BEGIN_OUTPUTS_DEFINITION(MAPSCOMPONENTTEMPLATE)
   //MAPS_OUTPUT("oName",MAPS::Integer32,NULL,NULL,1)
MAPS_END_OUTPUTS_DEFINITION

// Use the macros to declare the properties
MAPS_BEGIN_PROPERTIES_DEFINITION(MAPSCOMPONENTTEMPLATE)
   //MAPS_PROPERTY("pName",128,false,false)
MAPS_END_PROPERTIES_DEFINITION

// Initialization: Birth() will be called once at diagram execution startup.
void MAPSCOMPONENTTEMPLATE::Birth()
{
   ReportInfo("Passing through Birth() method");
}
// Core() is called whenever data is present
void MAPSCOMPONENTTEMPLATE::Core()
{
   ReportInfo("Passing through Core() method");
   // Sleeps during 500 milliseconds (500000 microseconds).
   Wait(500000);
}
// De-initialization: Death() will be called once at diagram execution shutdown.
void MAPSCOMPONENTTEMPLATE::Death()
{
   ReportInfo("Passing through Death() method");
}
Automatically Generated RTMaps Components

- Simulink .tlc file provided to generate RTMaps components directly from Simulink
Thanks for listening!

© Copyright 2017, dSPACE Inc.
All rights reserved. Written permission is required for reproduction of all or parts of this publication.
The source must be stated in any such reproduction.
This publication and the contents hereof are subject to change without notice.
Benchmark results are based on a specific application. Results are generally not transferrable to other applications.
Brand names or product names are trademarks or registered trademarks of their respective companies or organizations.
Embedded Success dSPACE