Video and Image Processing Embedded System Design

Houman Zarrinkoub, PhD.

Marketing Manager

The MathWorks

MathWorks
Aerospace and Defense Conference '07

MathWorks Aerospace and Defense Customers

Video and Image Processing Application Examples

Autonomous Vehicles

Night Vision

Targeting

Surveillance

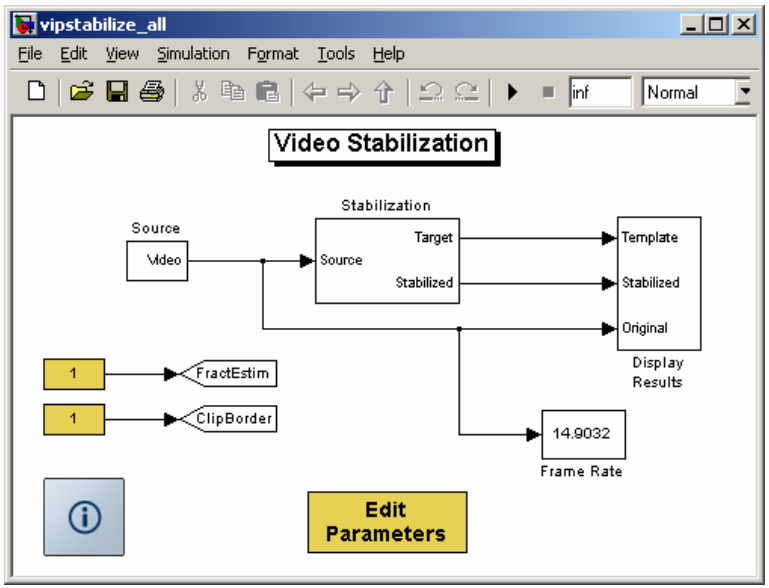
FLIR

Heads-Up Display

Satellite

Border Security

Demo: Video Stabilization



Traditional Workflow

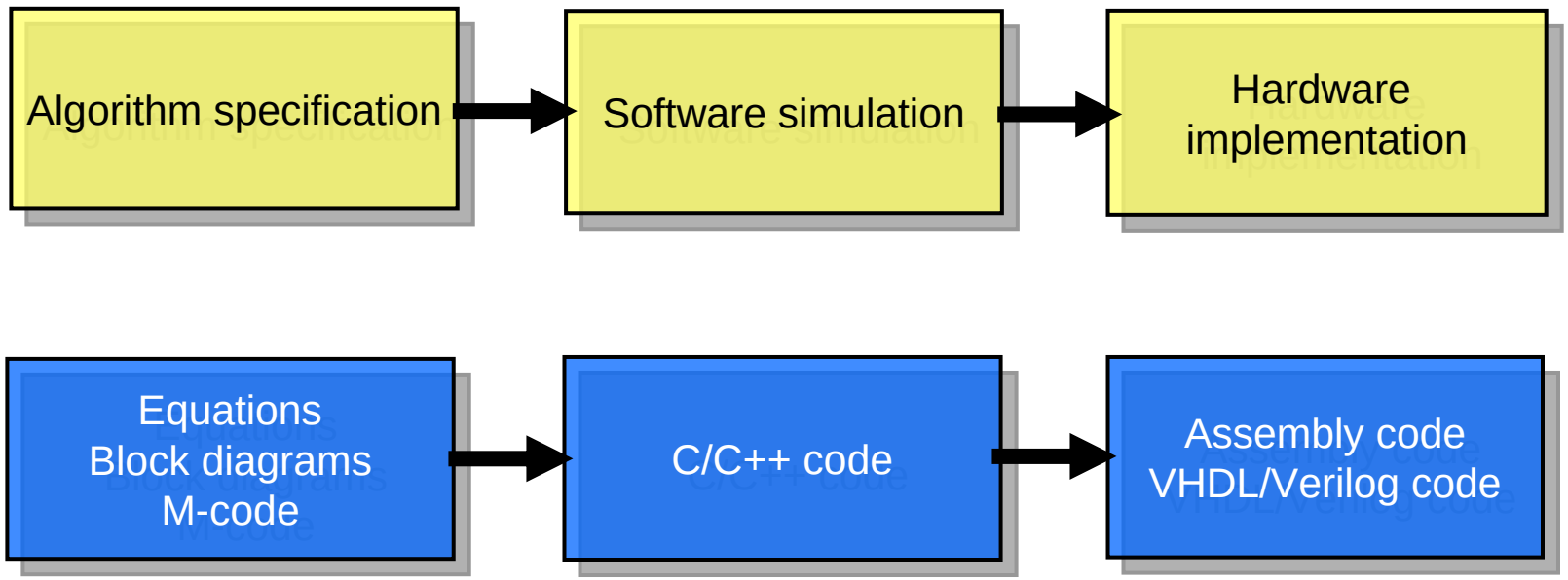
1. Develop floating-point algorithm — *MATLAB*
   - Focus on algorithmic integrity, proof of concept
2. Simulate (floating-point)
   - Iterate on algorithmic trade-offs
   - Validate against requirements

3. Convert design to fixed-point — *C/C++*
   - Focus on viability within implementation constraints
4. Simulate (fixed-point)
   - Iterate on implementation trade-offs
   - Validate against original requirements
5. Generate code for implementation

6. Validate and verify design after deployment — *Assembly or HDL*

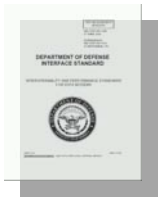
Multiple Truths in Traditional Workflows

Re-implement as you go down the level of abstraction



Model-Based Design with Simulink

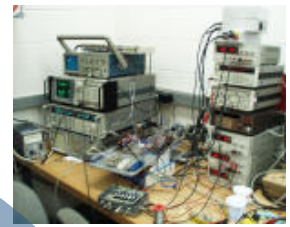
Requirements and specifications



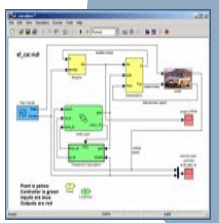
Design

Implementation

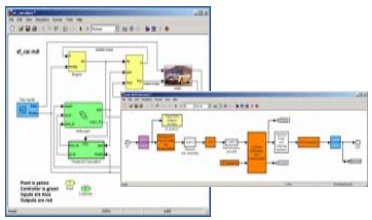
Test and verification



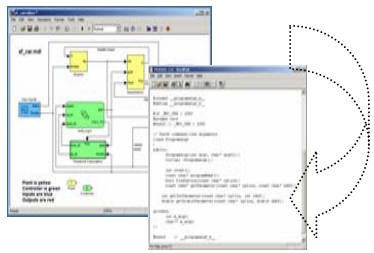
Model elaboration



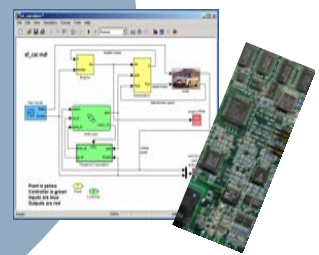
Executable specifications



Design with simulation



Automatic code generation



Continuous verification

Video and Image Processing Blockset

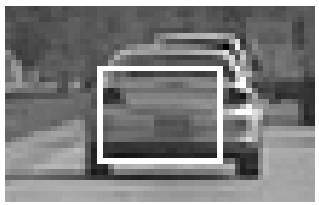
- Model, simulate, implement, and verify real-time video and imaging systems
  - Includes over 60 components and 100's of algorithms
  - Ideal for implementation of embedded systems

Streaming Video in/out | Detection, Thresholding | Tracking, Counting | Background Estimation

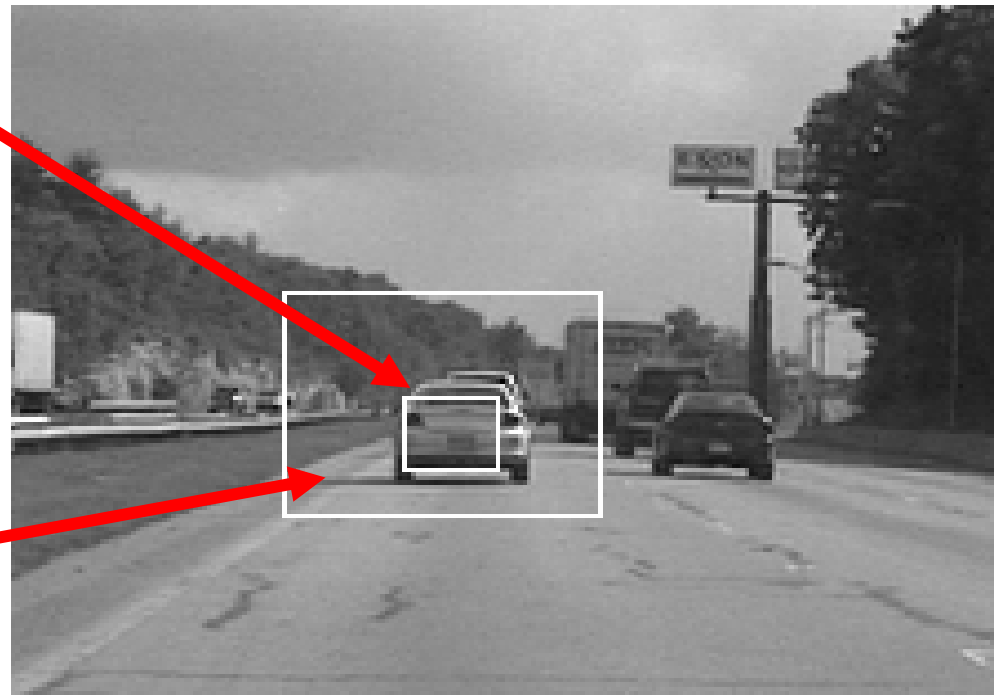
Demo: Video Stabilization

Goal: track license plate area

Initial target template region

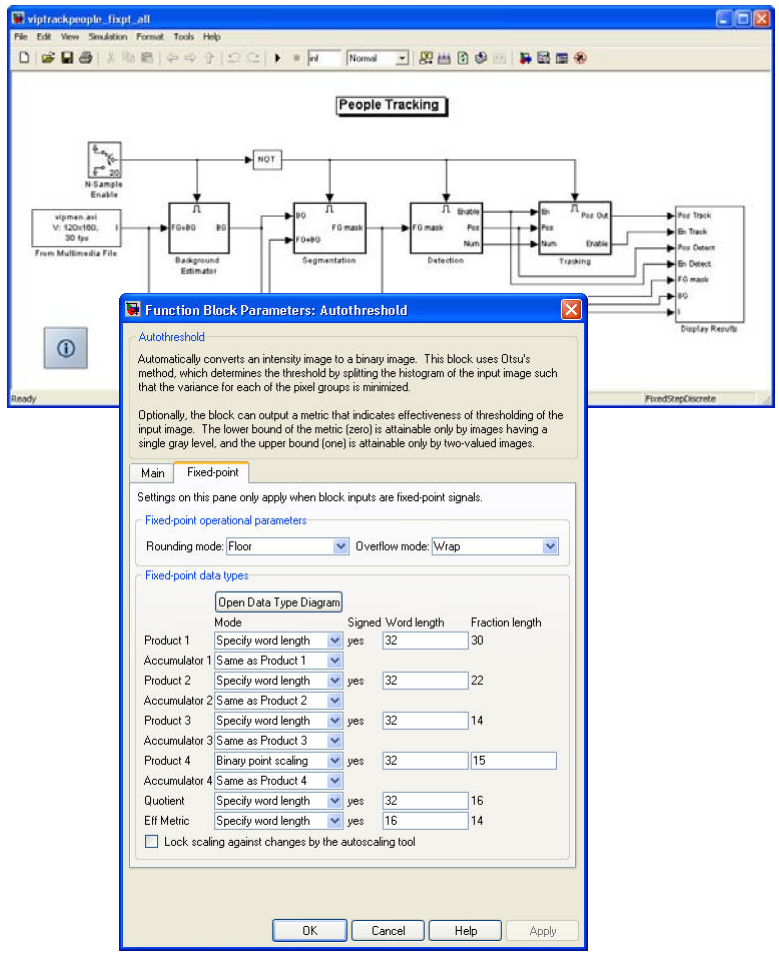


Motion search region



Fixed-Point Modeling*

- Avoid inaccurate results due to finite word effects
- Access tools built directly into block interfaces
- Easily change parameters to model the impact of rounding, overflow, and scaling



**** Requires Simulink® Fixed Point for integer and fixed-point data types***

C Code Generation and Co-Simulation

- C code generation from Simulink
  - Real-Time Workshop®
  - Real-Time Workshop® Embedded Coder
- Co-simulation of C code in Simulink
  - Link for Code Composer Studio™
  - Link for TASKING®
- Embedded targets from Simulink
  - Target for TI C6000™

HDL Generation and Co-Simulation

- Synthesizable VHDL and Verilog from Simulink
  - *NEW* Simulink HDL Coder (MathWorks)
  - Filter Design HDL Coder (MathWorks)
  - System Generator for DSP™ (Xilinx)
  - DSP Builder (Altera)
  - Synplify DSP (Synplicity)
- Co-simulation of VHDL and Verilog in Simulink
  - Link for ModelSim® (MathWorks)
  - *NEW* Link for Cadence® Incisive® (MathWorks)
  - ModelSim Xilinx Edition (Xilinx)
  - ModelSim Altera Edition (Altera)

Advantages of Model-Based Design

- Maintain “One Truth”
- Work in an integrated environment
  - Visualization and analysis
  - Modeling and simulation for embedded design
  - Implementation and testing
- Perform rapid prototyping
- Reduce errors and improve time-to-market

Doheny Eye Institute Enables The Blind To See With MathWorks Tools

The Challenge

- To develop a retinal prosthetics proof-of-concept prototype that interfaces with a permanent micro-electronic retinal implant

The Solution

- Used Simulink, Blocksets, Real-Time Workshop, and Target for TI C6000 to build and validate a rapid prototype on a DM642

The Results

- Completed phase II of research project ahead of schedule
- Currently in trials with patients though Doheny Eye Institute at the University of Southern California

Example of the Prototype

"We are working on real-time image processing with the TI DM642 processor as the target. The Video and Image Processing Blockset makes the task of creating our design and working prototypes much simpler."

Dr. James Weiland
Director, Intraocular Retinal Prosthesis Lab

Thank You