

Particle Filter based SLAM

Aswin P Ajayan

Indian Institute of Technology, Bombay

Jun 30, 2020

Overview

- 1 What is SLAM
- 2 Objective
- 3 Types of SLAM
- 4 Markov assumption and Recursive Bayesian estimate
- 5 Frameworks for recursive filter estimation
- 6 Particle Filters
- 7 Camera Driver
- 8 Pin Configuration
- 9 FPGA Resources Used
- 10 Simulation Results
- 11 Future Work
- 12 References

What is SLAM

- Computing robot's poses and map environment simultaneously
- Localisation : estimating robots location
- Mapping : building a MAP
- Given
 - $u_{1:T} = \{u_1, u_2, u_3 \dots u_T\}$, the control inputs
 - $z_{1:T} = \{z_1, z_2, z_3, \dots, z_T\}$, observations
- Wanted
 - m , map of the environment
 - $x_{0:T} = \{x_0, x_1, x_2, \dots, x_T\}$, robot location
 - $p(x_{0:T}, m | u_{0:T}, z_{1:T})$, the SLAM posterior

Objective

- To review the literature available on SLAM. To get acquainted with various techniques available for SLAM and examine some of them in depth. Simulation in real life constraints using ROS.

Types of SLAM techniques explored includes

- **Kalman Filter based approaches**

- Extended Kalman Filter
- Unscented Kalman Filter
- Extended Information Filter

- **Particle Filter based approaches**

- Fast SLAM - Rao Blackwellised Particle Filter
 - Augmented MCL
 - Mixture MCL

Types of SLAM

- Full SLAM vs online SLAM
 - Full SLAM $p(x_{0:T}, m | u_{1:t}, z_{1:t})$
 - Full SLAM $p(x_t, m | u_{1:t}, z_{1:t})$
- Feature Based SLAM vs Grid Based



- Active SLAM vs Passive SLAM

Recursive Bayesian Estimation

- **Markov assumption** : next state depends only on the present state

$$p(x_t | x_{0:t}, u_{1:t}) = p(x_t | x_{t-1}, u_t)$$

- **Recursive Bayesian Estimation**

$$\begin{aligned} bel(x_t) &= p(x_t | z_{1:t}, u_{1:t}) \\ &= \eta p(z_t | x_t) \int_{x_{t-1}} p(x_t | x_{t-1}, u_t) * bel(x_{t-1}) dx_{t-1} \end{aligned}$$

we can split this into predict and update steps where

Predict Step

$$\overline{bel(x_t)} = \int_{x_{t-1}} p(x_t | x_{t-1}, u_t) * bel(x_{t-1}) dx_{t-1}$$

Update Step

$$bel(x_t) = \eta * p(z_t | x_t) * \overline{bel(x_t)}$$

Bayes filter recursive estimation is realised using kalman filter, Information Filter, particle filter etc.

Kalman Filter Family

Beliefs are represented in parametric form with mean vector μ_t and covariance matrix Σ_t . Closed form expressions are available for belief propagation under certain cases

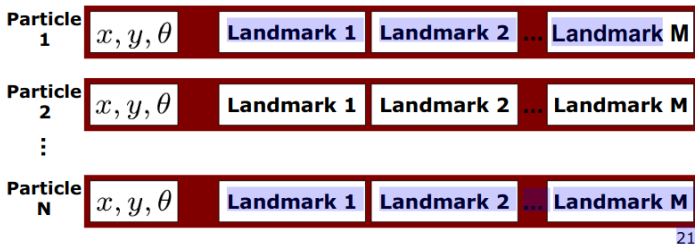
- **Kalman Filter** : Assumes linear models (state transition and measurement) and gaussian posterior
- **Extended Kalman Filter** : Linearises the model and then applies normal KF
- **Unscented Kalman Filter** : Samples a gaussian posterior from the resulting non linear transformation
- **Information Filters** : Uses canonical representation of Belief i.e. μ_t^{-1} and Σ_t^{-1}

Particle Filter

- This is a long Wave Infrared Focal Plane Array(IRFPA), 384X288, 25 um Uncooled Microbolometer. Spectral range of 8um 14um.
- DLC384 works in rolling shutter mode, meaning integration of one row at a time and simultaneous reading of previous row.
- We will see what is IRFPA and ROIC to better understand the device like DLC384, in general.
- **IRFPA** - Is an infrared(IR) image sensing device consisting of an array of IR sensing pixels placed at the focal point of a lense. This convert IR radiations to electrical signals. These IR sensing pixels can be a bolometer(in case of DLC384) which changes resistance if IR radiation is incident on it. IR detector are cooled to improve the signal to noise ratio (S/N) and to keep the temperature of the detector at a constant point in DLC384 thermoelectric cooling is used.
- **ROIC** - It is a silicon chip with circuit that integrate, sample and holds, amplify and multiplex the week detector current.



DLC384 Sensor Pin Diagram

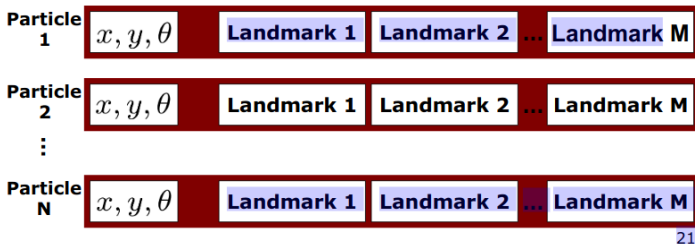


21

3

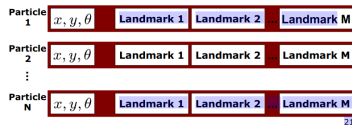
⁴ Source: DLC384 User Manual

Example circuit for signal pathway from one pixel to the output



21

4



21

⁵ Source: A Design of Readout Circuit for 384x288 Uncooled Microbolometer Infrared Focal Plane Array - 2012 IEEE 11th International Conference on Solid-State and Integrated Circuit Technology

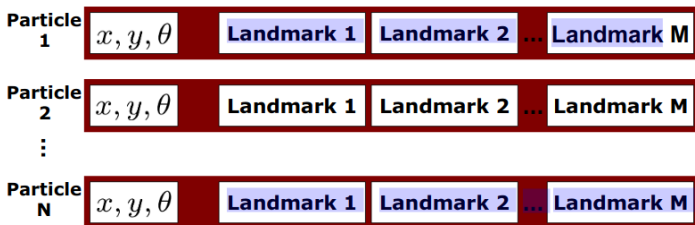
Working of IRFPA and ROIC using example circuit

- For each N^{th} -column there will be a blind pixel, a Integrator and sample and hold circuit, a output buffer (in rolling shutter mode). Output buffers are multiplex on Vout pin which is read by camera driver after digitization of its analog value.
- The induced small current because of change in resistance of Active Pixel by the IR radiation is integrated. This integrated voltage is stored in capacitor Cint. Usually this integration happen till the INT signal is high, coming from camera driver.
- Simultaneously previous row pixels value which is hold in capacitor Csh is read. Usually the reading of all the pixels of a row by camera driver happen till the VOUT_EN signal is high. This signal is set/reset by ROIC.
- After integration of all the pixels of a row and reading of all the pixels of previous row, VOUT_EN and INT are low, and the integrated voltage from Cint is sampled and hold to Csh.

Working of IRFPA and ROIC using example circuit (continue)

- A control unit is part of the ROIC which multiplex row and column data to VOUT.
- Integrator, sample and hold (S/H) circuit can be same for a column (in rolling shutter mode) or a part of unit cell (in Global shutter mode).
- While reading the voltage from Csh, next row (in rolling shutter mode) or next frame (in Global shutter mode) is integrated on "Cint".

Block diagram of IRFPA and ROIC for Rolling and Global shutter Mode



21

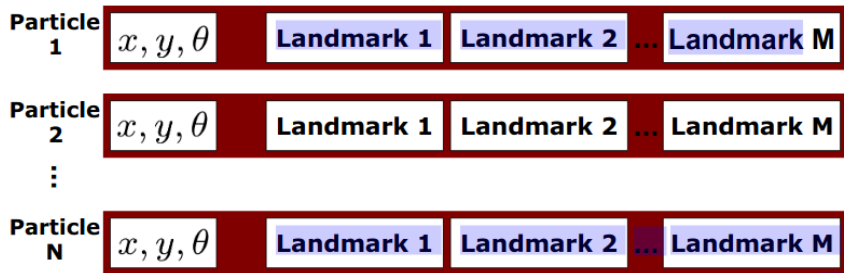
5

Source:

⁵A Design of Readout Circuit for 384x288 Uncooled Microbolometer Infrared Focal Plane Array - 2012 IEEE 11th International Conference on Solid-State and Integrated Circuit Technology

⁶Large dynamic range Readout Integrated Circuit for Infrared Detectors -2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)

Clock Diagram of DLC384



⁴ Source: DLC384 user manual

Explanation of DLC384 Clock Diagram

- Input to DLC384 from camera driver(which is implemented on FPGA)
- MCLK, INT, RST.
- Output from DLC384 to camera driver- VOUT-EN, FR,
VOUT(VOUT is a analog signal which is connected to camera driver
via n-bit ADC.
- DLC384 works in rolling shutter mode, meaning integration of one
row at a time and simultaneous reading of previous row.
- After RST is asserted from high to low, camera driver should set INT
signal to high for maximum of 384 cycles. RST reset the Readout IC
operation of sensor by forcing the integration of the signal on the first
row.
- After INT signal is asserted low (by camera driver), DLC384 will set
VOUT_EN and FR to high after exactly 18.5 cycles.
- FR trigger to high mark the start of the first row of a frame.

Explanation of DLC384 Clock Diagram (continue)

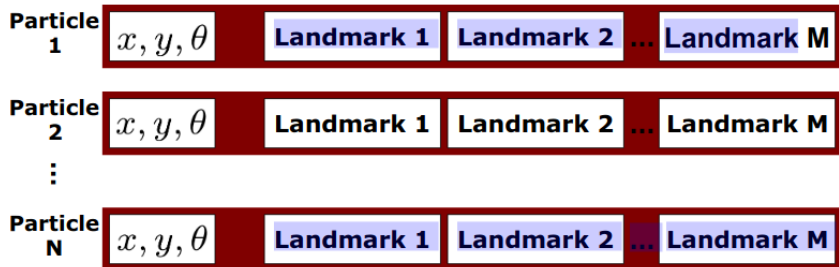
- From this point onwards pixel values of 1st row is available on VOUT, and on every rising edge of clock next pixel value is loaded on VOUT. Simultaneously INT should be asserted high by camera driver for maximum of 384 cycles so that integration of the signal on the next row can be done.
- Once all the pixels of a row are read (after 384 cycles), VOUT_EN will assert to 0 and after 18.5 cycles from INT falling edge it will assert to high so that next rows pixels can be read.

Camera Driver

Camera Driver have following Components -

- **Phase Locked Loop (PLL)**- Generated using the megafunction of quartus prime, generating 6.25 MHz so that 50 frames per second can be read from DLC384. Its input is from a 50 MHz onboard oscillator.
- **DLC384 reading FSM**- This on sensing inputs from IR sensor (VOUT_EN, FR) and Resetn from onboard switch, produces signals(INT, H_Counter, V_Counter) to read pixels value and store it to the Dual Port RAM.
- **Address Cal**- Using H_Counter, V_Counter this module calculate the index/address ($= \text{count_V} * \text{number of column} + \text{count_H}$) of the pixel. Output of this module is connected to Dual Port RAM read address line.
- **Dual Port RAM**- Generated using the megafunction of quartus prime. 8 bit word length and 2^{16} bits addressable. Simultaneously read and write operation can be performed, can be further interfaced with VGA Port
- **DLC384 Controller**- Configures the DLC384 Sensor

RTL of Camera Driver



Analysis of DLC384 Reading FSM Diagram and clock diagram from data sheet



⋮



21



⋮

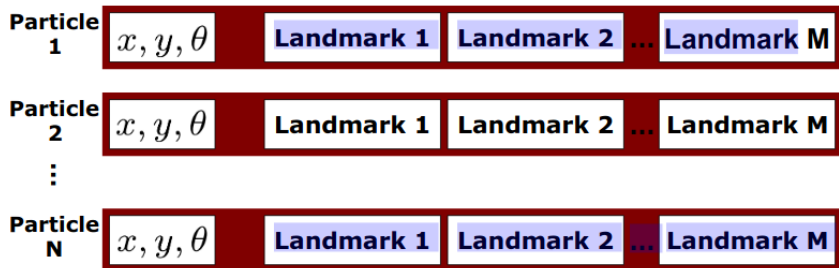


21

DLC384 Reading FSM

- DLC384 Reading FSM is a finite state machine having 4 states, a) State_Reset; b) State_INT; c) State_EN; d) State_Read.
- States (c) and (d) are two main states, in which FSM toggles maximum amount of time.
- In state (C) FSM waits for VOUT_EN signal from DLC384 which is expected to arrive after 18.5 cycles from INT falling edge.
- In State (d) INT signal is high(for 384 cycles) so that integration of next row pixels can be done. Also since VOUT_EN is high so simultaneously current rows pixels value is captured at every rising edge of clock.

DLC384 Controller FSM Diagram



DLC384 Controller

- There are two different input pins used to configure this sensor, SERIAL and SERDAT.
- If SERIAL = 0V then SERDAT is off and CTIA capacitance (on which integration charge accumulates) is fixed to 14 pF.
- If SERIAL = 5 V and SERDAT = 0V then CTIA capacitance is fixed to 18 pF or by sending some bit streams via SERDAT pin we can set different CTIA capacitance and different functions of the sensor can also be changed like(HFLIP,VFLIP).
- As can be seen in FSM diagram of the DLC384 controller, which transmit 51 bits from SERDAT pin if SERIAL pin is high.
- SERIAL pin is connected to DIP pin of FPGA board which can be turn on/off externally.
- Bit streams are hardcoded in FPGA which can be later controlled from DIP switches.

Pin Configuration and it's Key Number

In Port	Pin	Key	Out Port	Pin	Key
resetsn	j15	Key0	reset_out	PIN_A2	GPIO_02
inclk	R8	50 MHz osc	int	PIN_A3	GPIO_03
vout_en	PIN_D3	GPIO_00	clock_6.25	PIN_B3	GPIO_04
fr	PIN_C3	GPIO_01	pll_locked	PIN_A15	LED0
Vout_ADC	PIN_A4,B5,A5, D5,B6,A6,B7,D6	GPIO_06 to GPIO_013	serial_out	PIN_B4	GPIO_05
			serdat	PIN_A4	GPIO_06

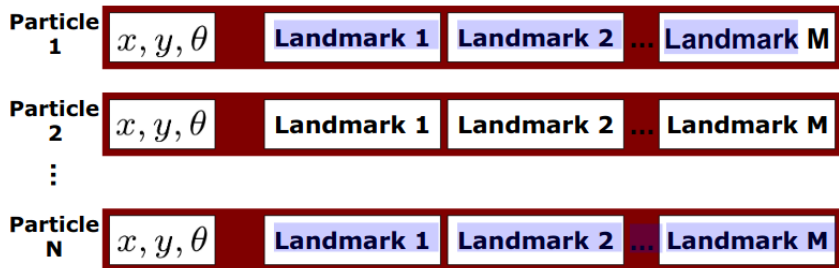
Table: 1

Resources Used

Resource Name	Number/% used
Total Logic Elements	153/22320 (less than 1%)
Total Registers	44
Total Memory Bits	524288/608256 (86%)
Total PLLs	1/4 (25%)

Table: 2

Simulation Results Part-1: Control Signal analysis



Simulation Results Part-1: Control Signal analysis

- To compare the simulated timing pulses of control signals with data sheet of DLC384 it is assumed that in a frame -
 - Number of columns are 5, in place of 384.
 - Number of Rows are 3, in place of 288.
 - Number of cycles after which the next row is read (VOUT_EN is asserted to 1 after INT is asserted to 0) is 2.5 in place of 18.5.
- Simulation timing pulses of control signals (figure-10) and Clock Diagram of DLC384 (figure-6) overlaps.
- we can calculate the pulse duration by looking at following counter values (In Figure-10).
 - For 2.5 cycle(for sensor it is 18.5), for which VOUT_EN is low, look for q_18_cycle.
 - For 5 cycle(for sensor it is 384), for which VOUT_EN and INT are high, look for s_h_counter.
 - For current s_v_count to be seen.

Simulation Results Part-1: Control Signal analysis (continue)

- Using `s_v_count` and `s_h_count` values current pixel location/index can be calculated.
- For reset, look for `s_reset_out`.
- 6.25 MHz clock is `s_clock_6_out`.
- Signal `fr` marks the start of a new frame, after reset is done.
- If "`s_serial`" is high then "`s_serial_out`" is set to high and at "`s_serdat`" configuration bits are sent.

Simulation Results Part-2: Working on actual images stored in a text file

- 3 images of same pixel size is taken, considered as 3 consecutive frames of a video.(img_1.png, img_2.png, img_3.png)
- Using a python script, images are converted into a single vector and then saved in a file one after another. This file will simulate the camera pixels.(Python script - image_to_text.ipynb , file - img_1_2_3.txt)
- In Quartus, a test bench is created which simulates the Camera sensor DLC384 sensor and read file img_1_2_.txt for throwing pixels value.
- Sensor DLC384 gives analog value at VOUT pin, so a ADC considered b/w the sensor and the camera driver.
- Once a frame is sent, testbench reads the Dual Port RAM and saves the data in another file output_img.txt.
- After the simulation both img_1_2_3.txt and out_img.txt files are compared using a python script, which shows both matches perfectly.

Simulation Results Part-3: DLC384 Controller

- DLC384 Controller simulation can be seen in figure-10, where s_serial_out and s_serdat should be connected to SERIAL and SERDAT pins of the DLC384 sensor.

Future Work

- Built an interface between Dual Port RAM(ping_pong concept can also be introduced) and VGA port, so the video can be observed on a VGA ,monitor.
- DeO-Nano development board give 3.3 V as high and 0V as low, but Sensor pins works at 5V(as high) and 0V(as low) so level-shifter circuit need to be used.

References

- [1] https://www.opli.net/opli_magazine/imaging/2017/sierra-olympic-introduces-world-first-true-hd-thermal-camera-jan-news/
- [2] <https://in.element14.com/terasic-technologies/p0082/e22f17c6n-DEO-nano-dev-kit/dp/2076463>
- [3] <http://www.123-cctv.com/CCTV-Monitors/7-inch-Security-LCD-Monitor-with-VGA.html>
- [4] DLC384 User Manual
- [5] A Design of Readout Circuit for 384x288 Uncooled Microbolometer Infrared Focal Plane Array - 2012 IEEE 11th International Conference on Solid-State and Integrated Circuit
- [6] Large dynamic range Readout Integrated Circuit for Infrared Detectors -2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)

Thank You for your Attention