ADVANCED IMAGE ANALYSIS

MAIA, 2016 – 2017, 2ND SEMESTER

UNIVERSITY OF CASSINO AND SOUTHERN LAZIO

HEART FLOW ANALYSIS IN ECG-GATED CARDIAC ULTRASOUND IMAGES

Benjamin LALANDE CHATAIN

Umamaheswaran RAMAN KUMAR

Yuliia KAMKOVA

CONTENTS

- Project Goals
- **Implementation Steps**
- **Problems & Solutions**
- Application Flow
- Results
- Further Enhancements/Improvements

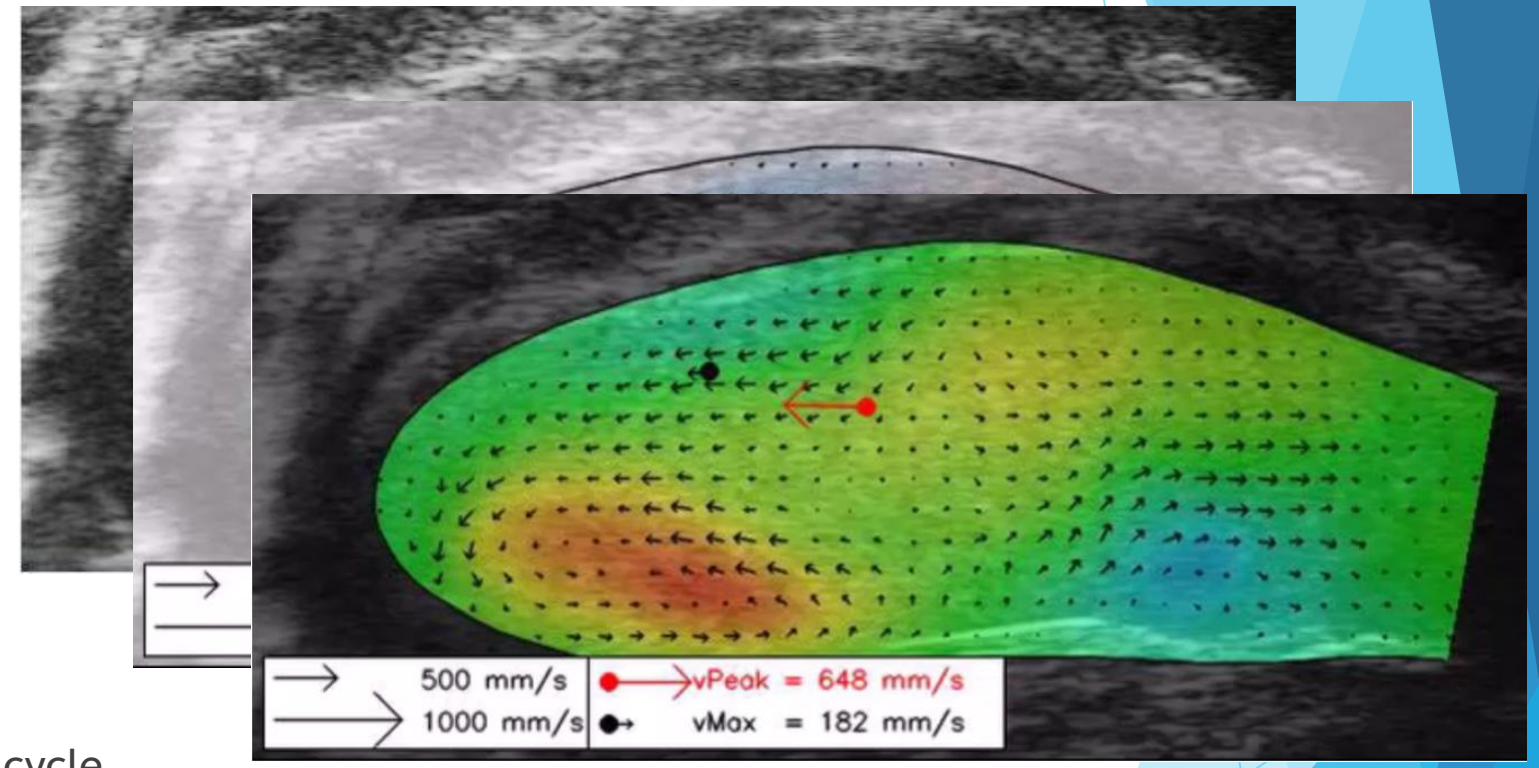
Project Goals

- ***Input : EKV Image***

- ***Qualitative Analysis***
  - Ventricle wall tracking
  - Velocity vector field
  - Alpha compositing

- ***Quantitative Analysis***
  - Instant maximum velocity
  - Peak velocity of whole cardiac cycle

- ***Vortex Detection(Bonus)***



Implementation Steps

- ***Dense Flow***
  - Farneback method

    OpenCV - *calcOpticalFlowFarneback(…)*
  - Denseflow image -> HSV image

    OpenCV - *split(…)*

    Magnitude = $\sqrt[2]{{f_x}^2 + {f_y}^2}$, Angle = $\tan^{-1}\frac{f_y}{f_x}$

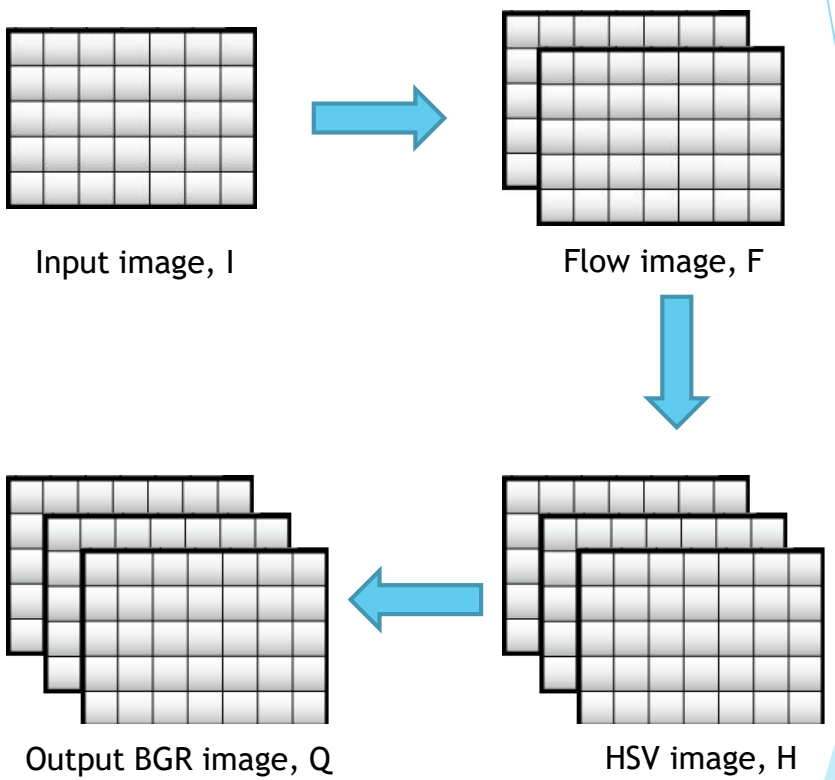
    H <- Angle $0 \leq H \leq 360$

    S <- Magnitude $0 \leq S \leq 1.0$

    V <- 1.0
  - Convert HSV->BGR

    OpenCV - *cvtColor(…, CV_HSV2BGR)*
  - Draw velocity field vector arrows

    OpenCV - *cv::arrowedLine(…)*



Implementation Steps(Cont.)

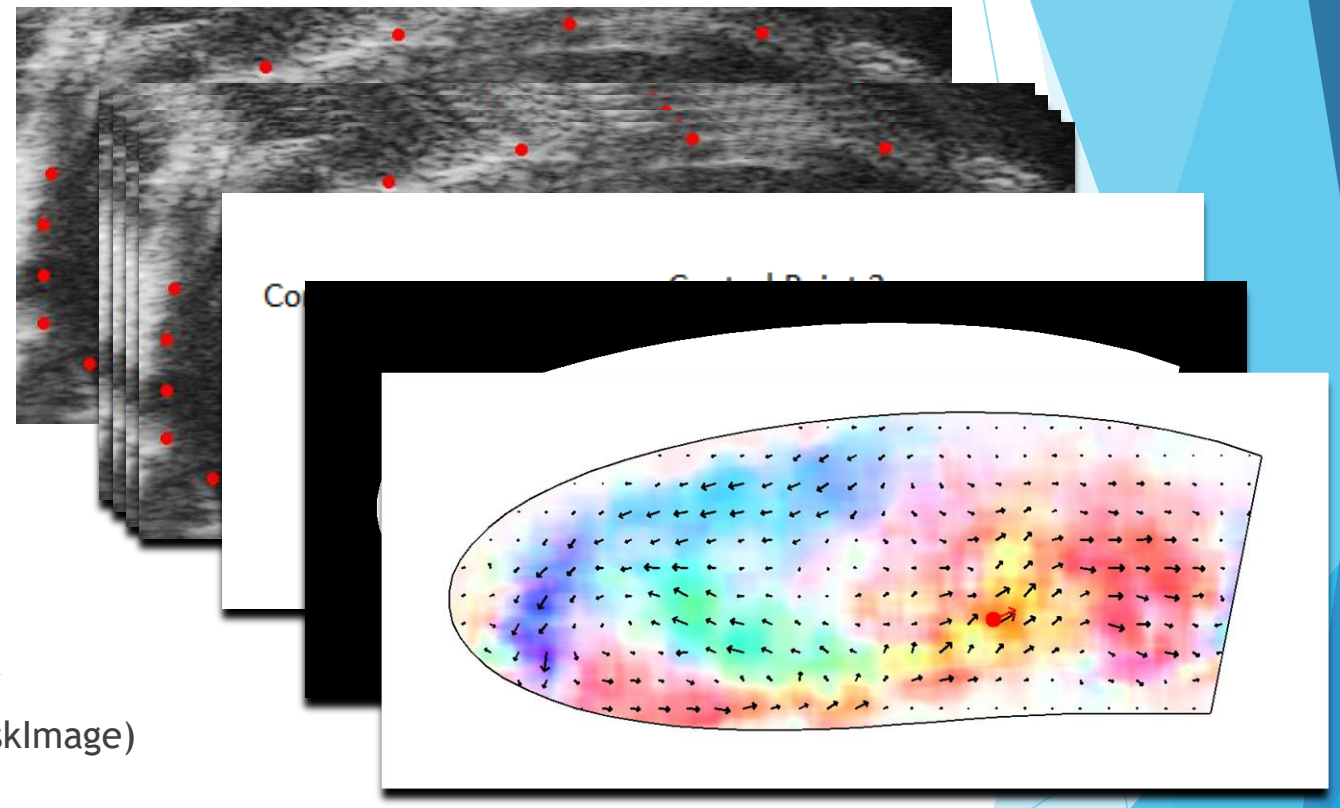
- ***Wall Tracking***
  - Select ventricle wall points

    Qt - *mousePressEvent*(…)
  - Lucas-Kanade method

    OpenCV - *calcOpticalFlowPyrLK(…)*
  - Bezier curve
  - Create mask image with contour

    OpenCV - *drawContours(…)*
  - Mask the output image with the mask

    OpenCV – inImage.copyTo(outImage, maskImage)



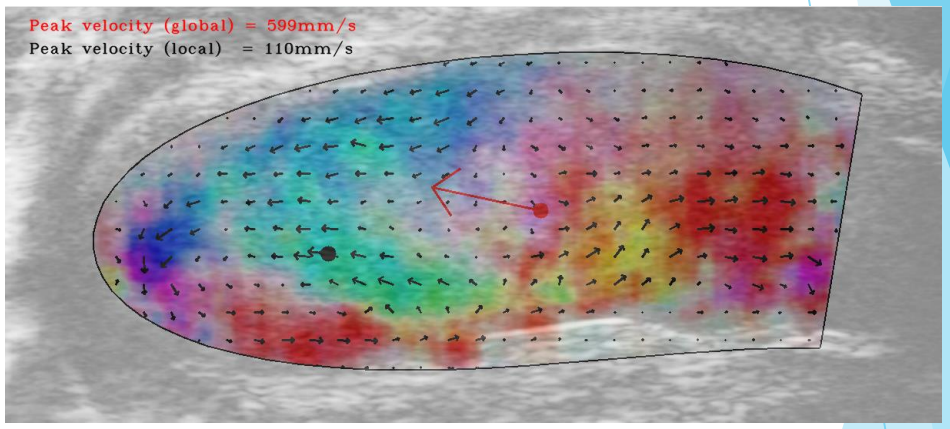
Implementation Steps(Cont.)

- ***Alpha Blending***
  - Convert input grayscale image to BGR
  - Blend input image with quantitative image

    OpenCV - *cv::addWeighted(…)*

- ***Maximum Peak Velocity***
  - $v_{peak} = \frac{magnitude_{max} * 0.015 * 1000}{0.2}\ mm/s$
  - Write text on final image

    OpenCV - *cv::putText(…)*



Implementation Steps(Cont.)

- ***Vortex Detection***
  - Poisson equation

    $$\nabla^2\psi = -\omega$$

  - Vorticity(Curl of velocity)

    $$\omega = \nabla \times v = \frac{\delta f_y}{\delta x} - \frac{\delta f_x}{\delta y}$$

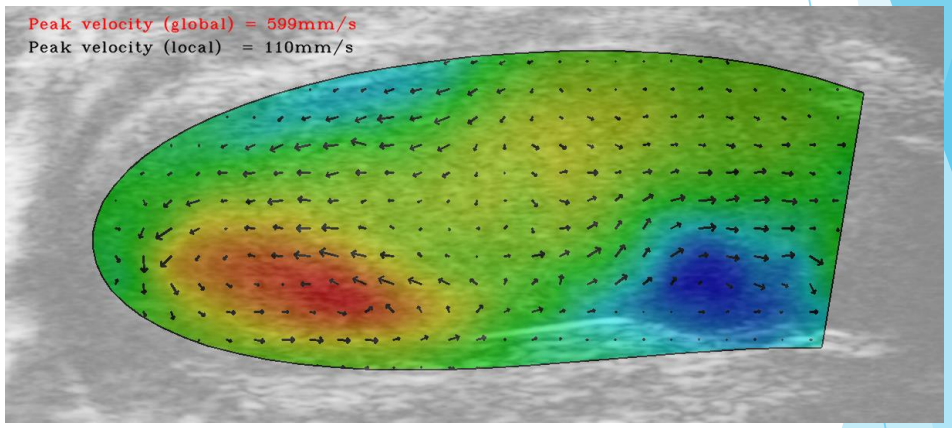
  - Derive f_y along x and f_x along y and subtract

    OpenCV - *Sobel(…)*
  - Solve for ψ using Laplace transform
  - Encode ψ to HSV image

    H <- $\begin{cases} 0 - 120 & if\ \psi \geq 0 \\ 120 - 240 & if\ \psi < 0 \end{cases}$

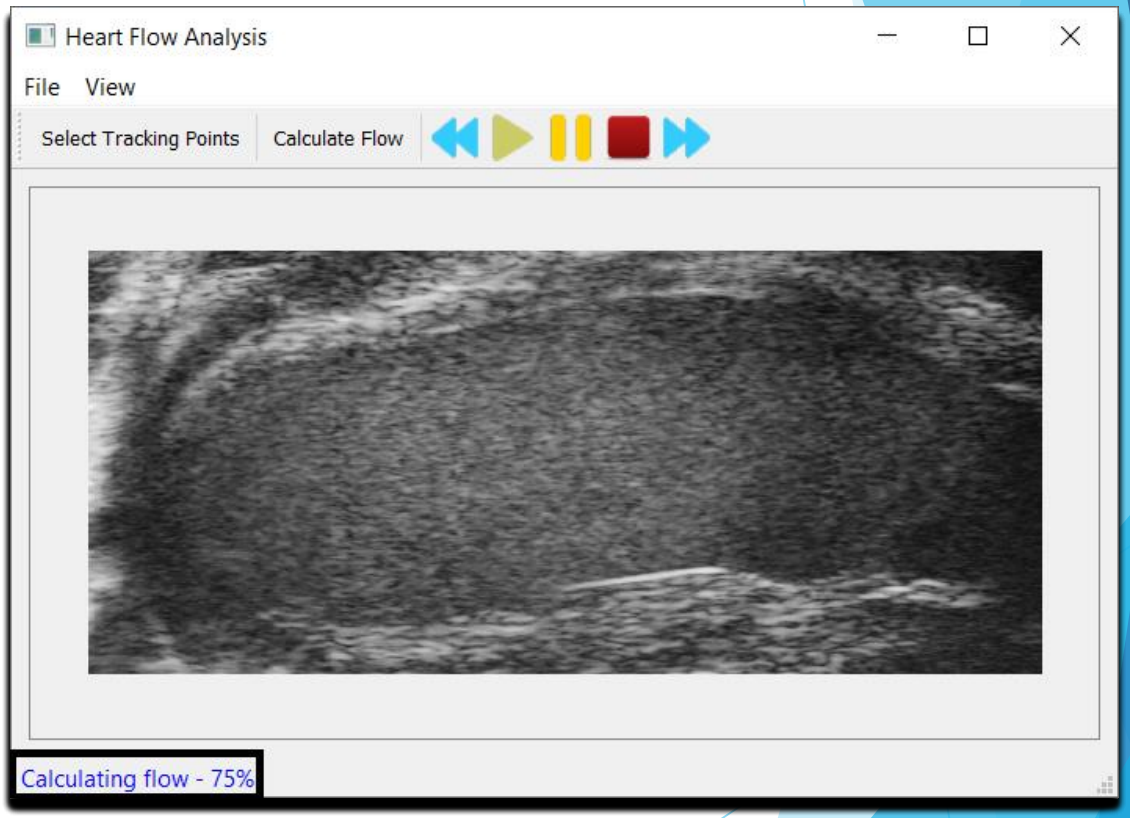
    S <- 1.0

    V <- 1.0



Implementation Steps(Cont.)

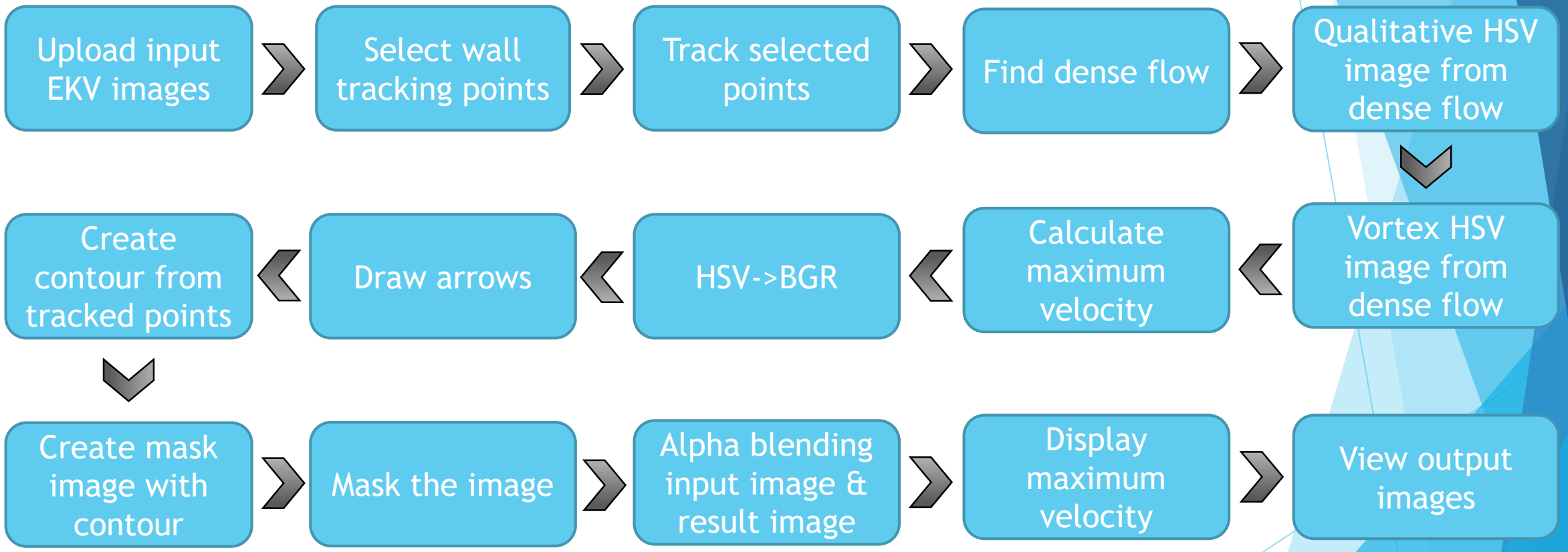
- ***User Interface***
  - Created with Qt gui components
  - *Menu bar*
    - Import input EKV images
    - Export qualitative & vortex images & videos
    - View both input and output set of images
  - *Toolbar*
    - Select tracking points
    - Start optical flow calculation
    - View all images as video
  - *Status bar*
    - Display status of images processed



Problems & Solutions

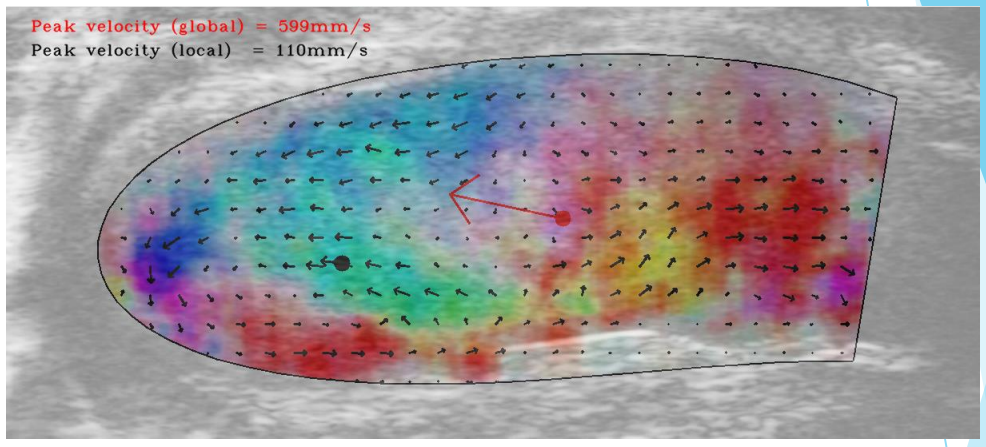
- Insufficient memory
  - Manually free memory whenever possible
  - Calculate display images on the fly
  - Compiler option to extend memory
- Application goes to ‘Not Responding’ state due to huge computation
  - Multi-threading
  - QCoreApplication::processEvents()
- OpenCV Mat & Qt QImage
  - Separate utility class to convert Mat to QImage

Application Flow

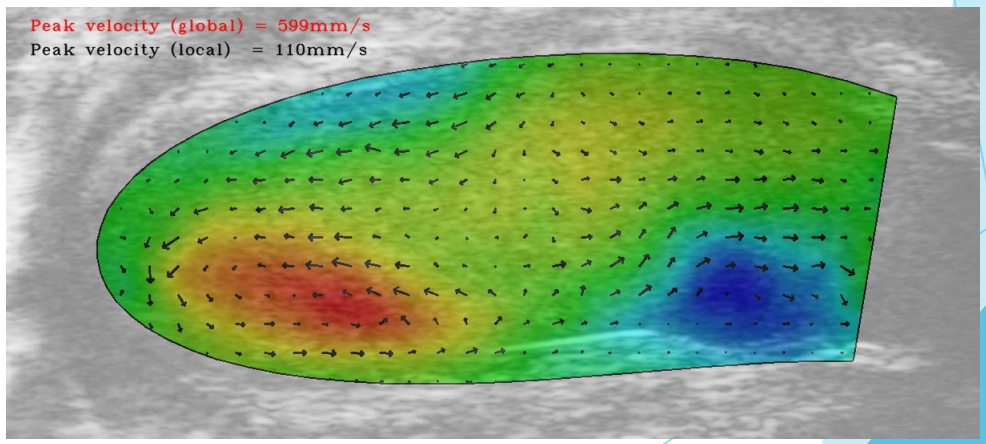


Results

- Qualitative & Quantitative Analysis ✓
  - Maximum peak velocity -> 599 mm/s



- Vortex Detection ✓



Further Enhancements/Improvements

- Image colour normalization
- Selection of tracking points on any image
- Memory management
- Qualitative analysis only within ventricle wall
- Slider bar for viewing images

THANK YOU