Dynamic fusion

大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

Internship Week 21-22
Segmented Fusion
19th July 2017

Last meeting

- Previously
  - Local to Global transform in CPU
  - Volume transform and projection in CPU
  - Then did it in GPU
- Plan for today's meeting:
  - Read papers : conceive algo
  - Test for each block of code
  - Segmented fusion

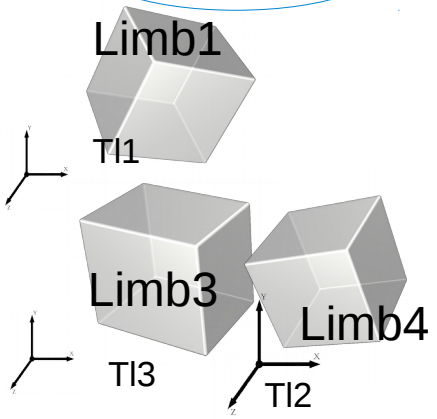
Segmented Fusion Pipeline

For each part

0 Bounding Box (PCA)

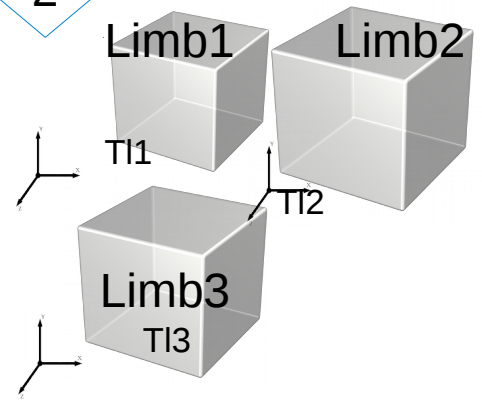
Compute dimensions

1 Create Volumes



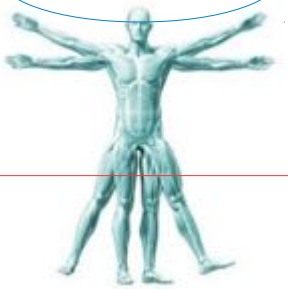
Local to Global coordinates

2



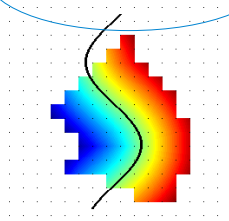
Transform in current pose

3 Align Pose

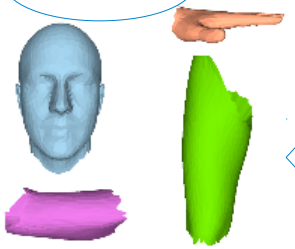


Projection in current image

4 TSDF Fusion

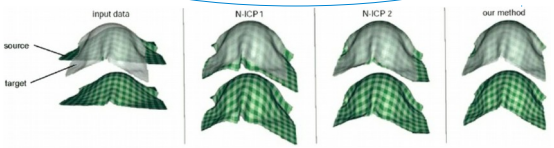


5 Mesh



6 Assembling mesh

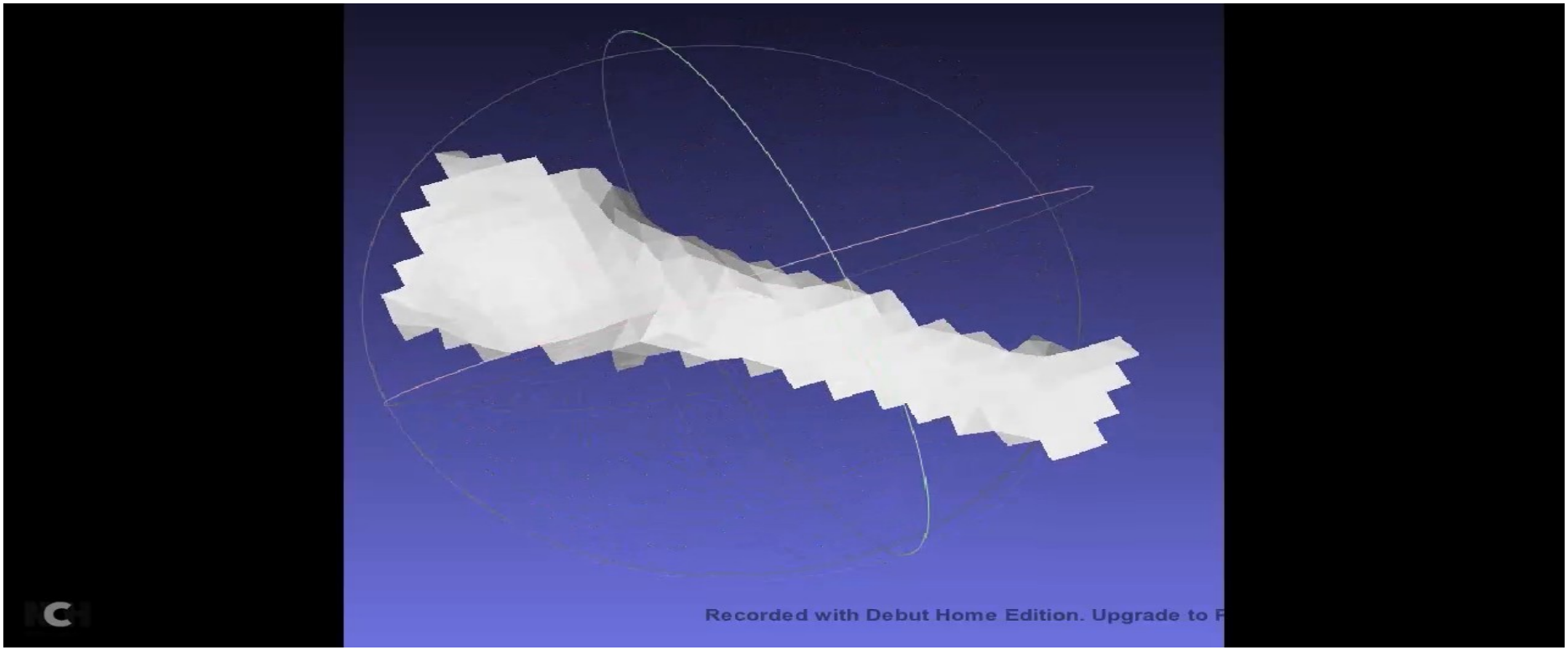
7 Rigid Alignment



End of 1

Marching cubes results

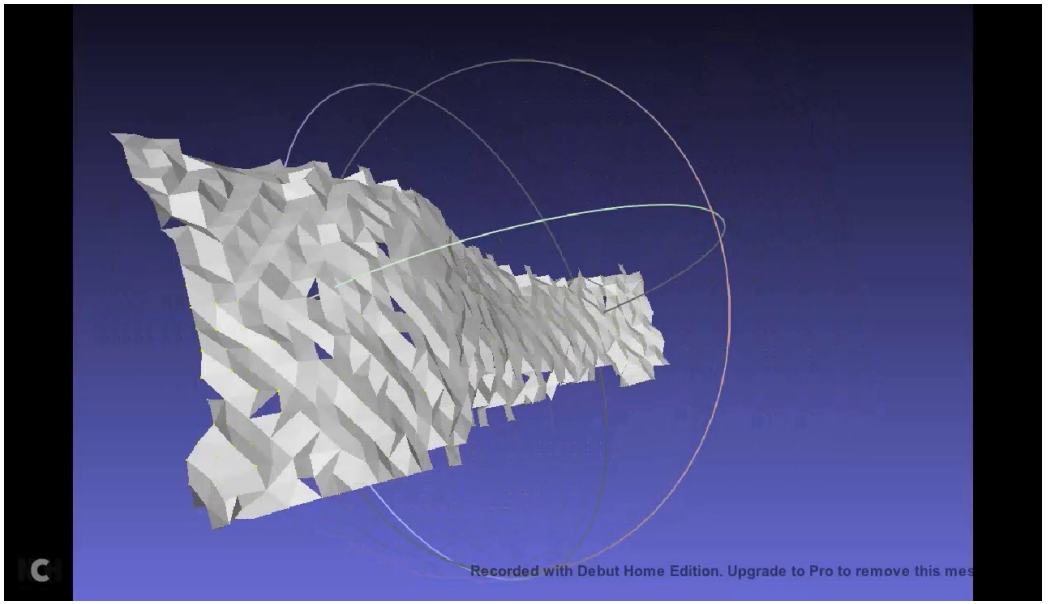
What I want for the left lower arm



Number of Vertices : about 200

Marching cubes results

What I got



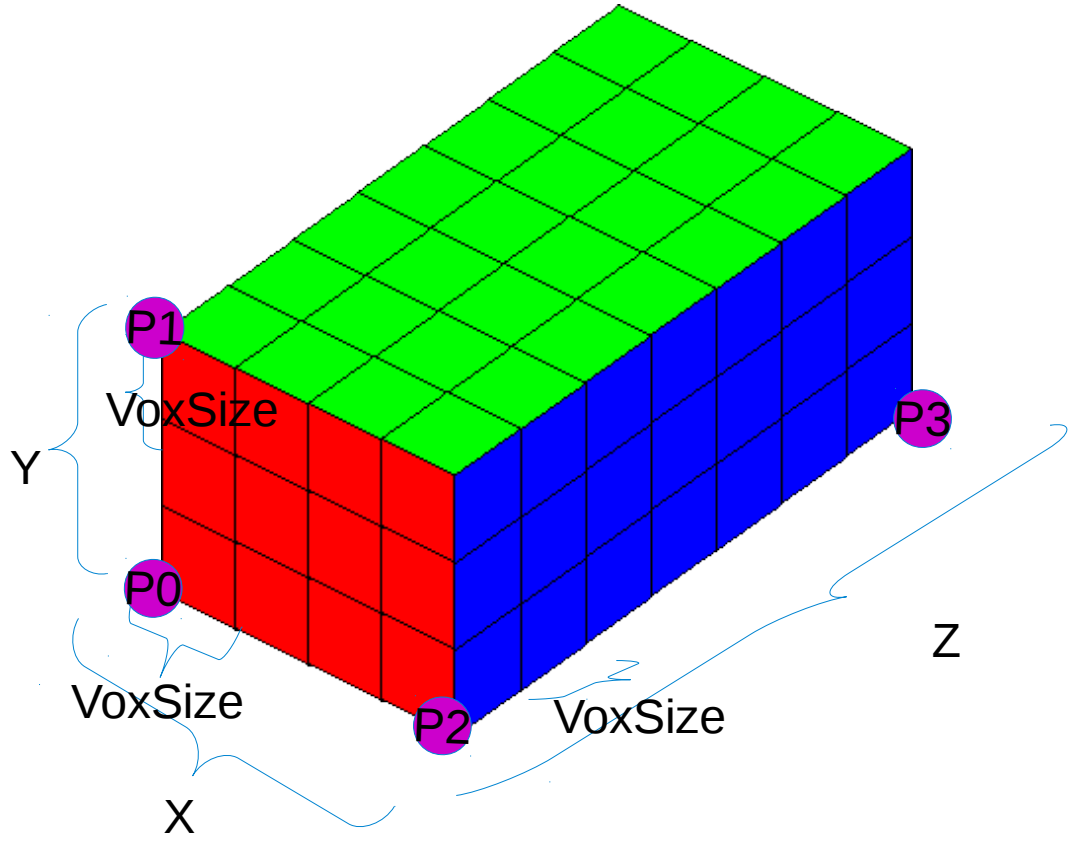
Algo (same as before):
1) Compute dimension and scaling
2) Get transfo local to global using eigen vector + center of cloud of points in Global coordinates
3) Rescale
4) Transform to global coordinate

Quality not perfect but seems okay to continue

Better quality if sampling?

Volume length depth:

Compute the length of X, Y and Z



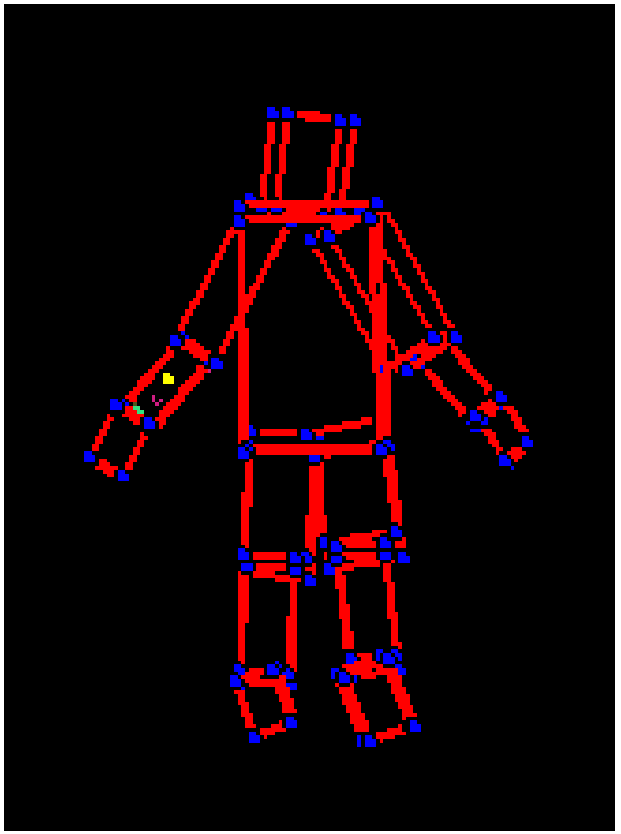
X = norm(P2 – P0)/VoxSize
Y = norm(P1 – P0)/VoxSize
Z = X

Resolution = 1/VoxSize

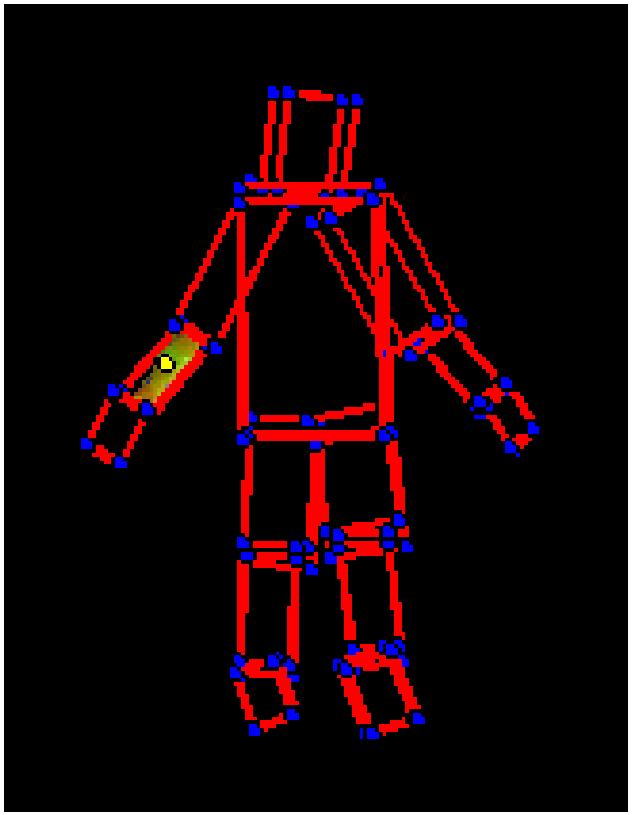
VoxSize = 0.0046

Volume length depth:

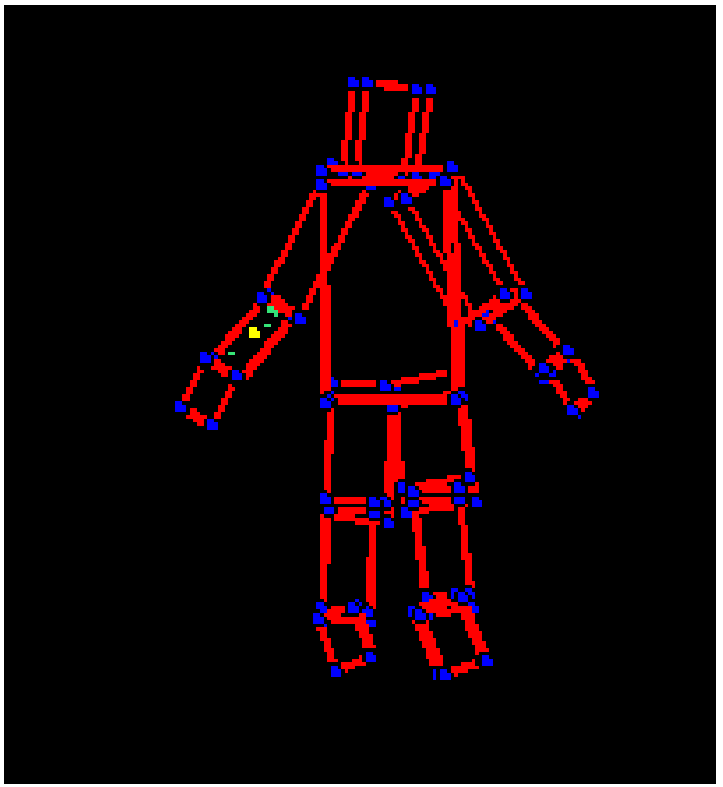
Z = X-1



Z = X

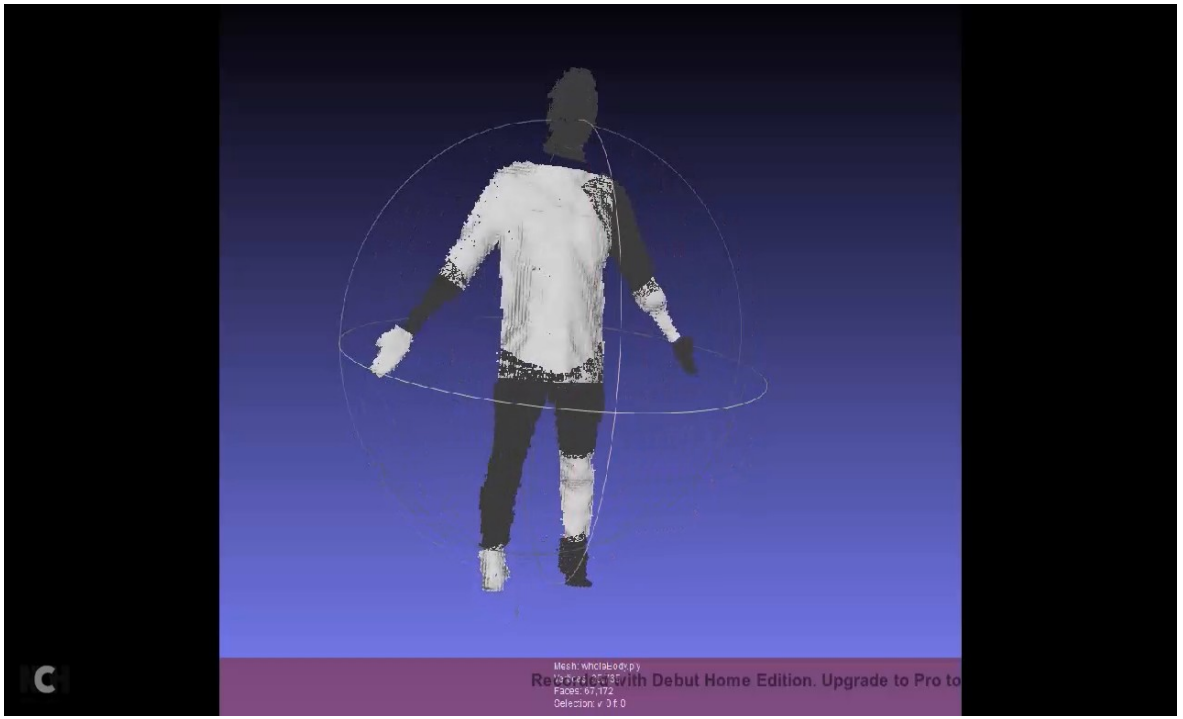


Z = X+1



Naive Stitching results

The result of the arm could be generalized on all limbs



For each body part after the TSDF and the Marching cubes :
1) Transform to global coordinate system the vertices and the normales
2) Concatenate the list of vertices and normales with the previous body part
3) Add the max value of faces of the previous body parts in the faces of the current face

Conclusion :Overlay seems okay for one lone image

Need of shoulder segmentation?

Segment tracking results

Tracking it from image 0 to image 19

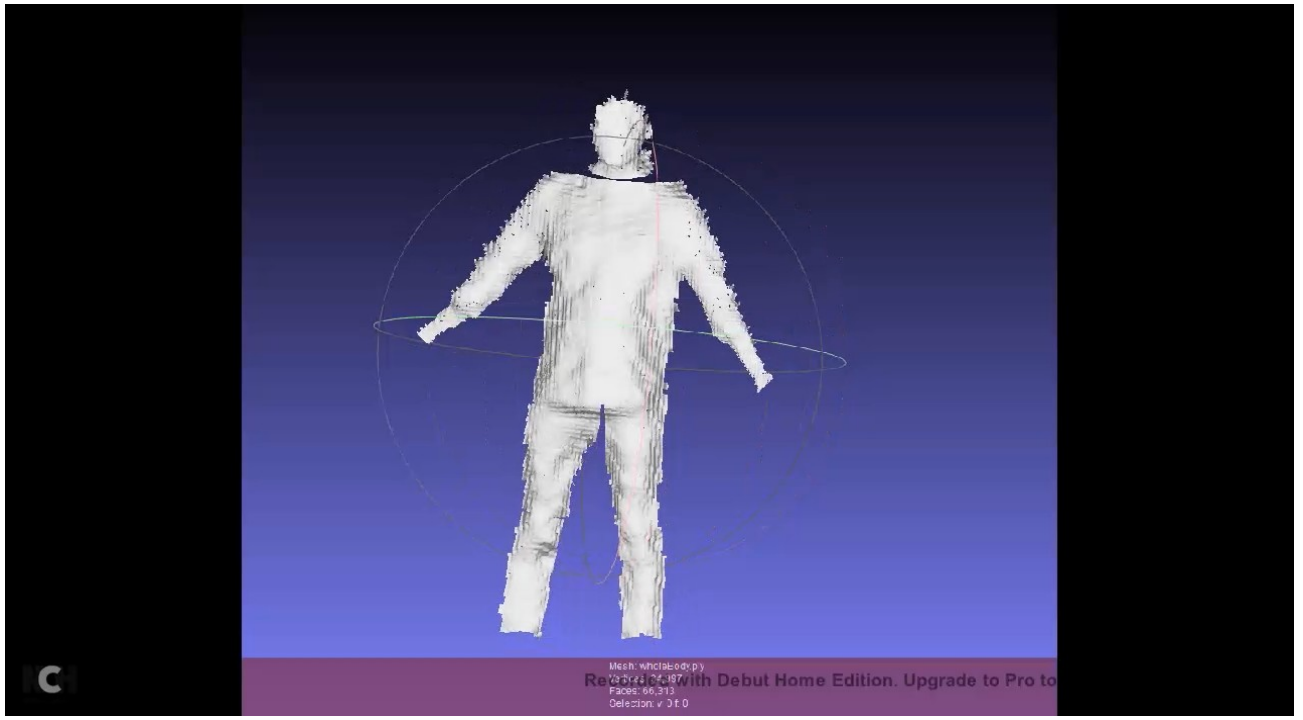
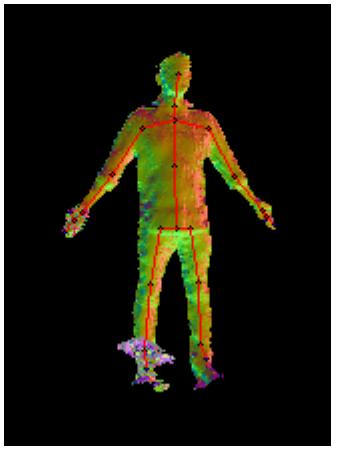


Image 30



Some image do not recover properly hands and feet. Therefore no hands and feet for now.

Tracking seems good for very small transformation
Time : about 6 minutes

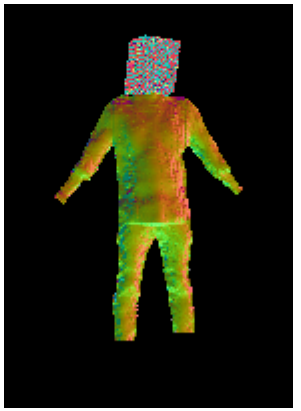
Segment tracking results

Algorithm :
1) Init: Native stitching => Generate mesh of whole (segmentation, transfo, fusion, marching cubes, stitching)
2) Get New Image
3) Get all the labels of the body
4) Tracking New image with stitched mesh
5) Native stitching (adding pose transfo)
6) Get back to 2)

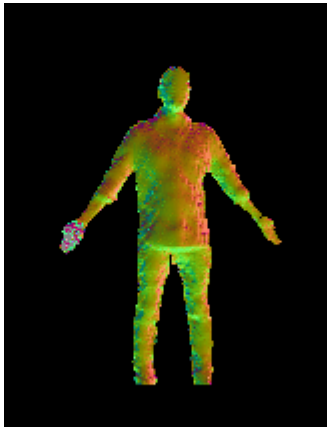
Some default are being ignored through tracking but others remains

Default depend on VoxSize

Image 12

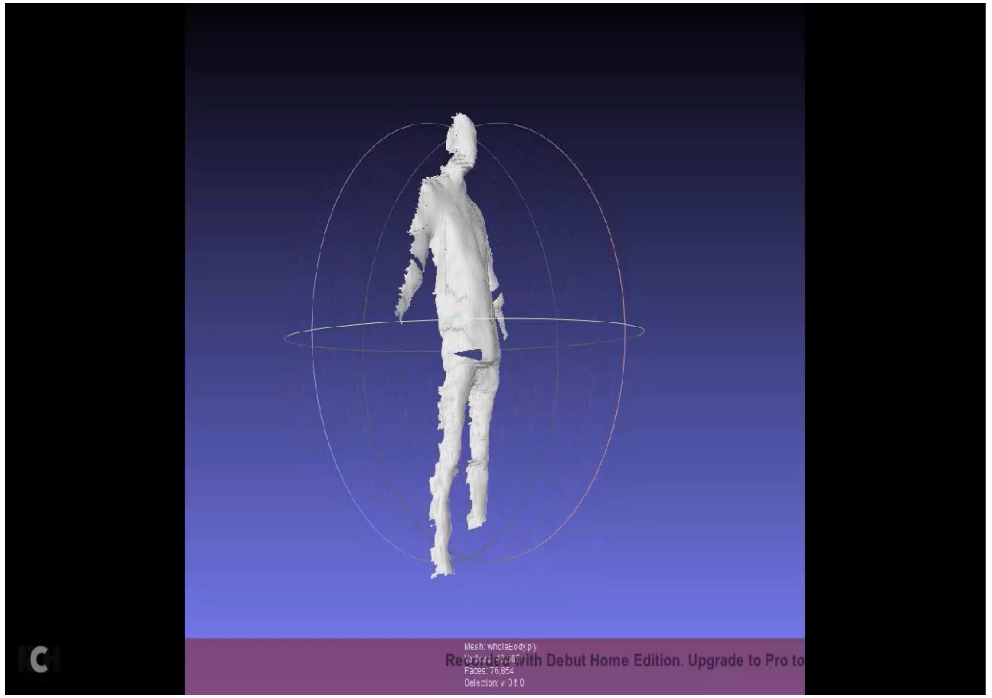


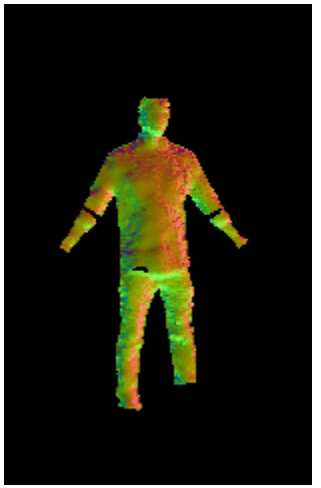
Full body Image 0-19



Segmented tracking results

In case of big leap : Tracking image 22-23





Naive Stitching should be improved!

Stitching Algo

Algo pipeline :
1) Define interface between two adjacent body parts
2) Define stitching weight for interface.
3) Transform each body part (local + align pose)
4) Transform for interface using orthogonal Procrustes problem
5) Reconstruct whole body
6) Re-align body parts interface by minimizing energy

Stitching Puppet explanation

- Goal : Estimate human shape and pose from 3D data
- Graphical model : create node = random variable **x** = [center,rotation vector,pose-dependent deformation,intrinsic body shape]
- Interface: “For neighboring body parts, we duplicate the vertices that are in common, creating a set of “interface points” that should match when the parts are stitched together.”

Stitching Algo : interface

Interface: duplicate points between two adjacent body part.

My first Idea was : add length to bounding boxes, duplicate vertices that are in two bounding boxes.

Stitching Algo : Stitching weight

Stitching weight

body parts to be loosely connected (Fig. 5); cf. [34]. We define the stitching potentials as a weighted sum of squared distances between the interface points of adjacent parts: $\Psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) =$

$$\exp(-\frac{1}{\|\mathbf{u}_{ij}\|_1} \sum_{k=1..N_{ij}} u_{ij}(k)\|\tilde{\mathbf{p}}_{i,I_{ij}(k)}(\mathbf{x}_i) - \tilde{\mathbf{p}}_{j,I_{ji}(k)}(\mathbf{x}_j)\|^2).$$

There are N_{ij} interface points between part i and part j. Let $I_{ij}(k)$ and $I_{ji}(k)$ denote the index of the k-th interface point on part i and part j, respectively. $\tilde{\mathbf{p}}_i(\mathbf{x})$ indicates the part points in the global frame after applying parameters $\mathbf{x}$ for each part. $u_{ij}(k)$ defines a stitching weight that is set to 0.8 for points that are allowed to stretch more, like the front of the knee or the back of the elbow, and is 1.0 otherwise; $\mathbf{u}_{ij}$ is a vector of these weights.

Stitching Algo : Interface transformation

Orthogonal Procrustes Problem on limbs transfo

Input : Two dimensionally identical matrix A and B
Goal : Approach as mush as possible the first matrix to the second one by computing an orthogonal matrix

$$R = \arg\min_{\Omega} \|\Omega A - B\|_F \quad \text{subject to} \quad \Omega^T \Omega = I$$

Output : Orthogonal matrix R that will be used for interface stitching transformation

$$R = UV^T$$

Where U and V are left-singular vectors and right-singular vector respectively of $M = BA^T$

Stitching Algo : Alignment

Alignment in stitching puppet is optimizing energy

$$E(\mathbf{x}, S) = E_{\text{stitch}}(\mathbf{x}) + E_{\text{data}}(\mathbf{x}, S),$$

S : 3D Mesh

Stitching term. The stitching term is the cost for disconnected parts, plus a penalty for penetration, $E_{\text{stitch}}(\mathbf{x}) =$

$$\sum_{i=0..15} \sum_{j \in \Gamma(i)} \alpha_{ij}(-\log(\Psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) + Q_{ij}(\mathbf{x}_i, \mathbf{x}_j)),$$

where $\Psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ is the stitching potential and Q_{ij} is a penalty for intersecting parts. For the latter, we use a simple test on the location of the part centers, setting Q_{ij} to zero if $||\mathbf{o}_i - \mathbf{o}_j||_2$ is more than 0.05, and 1.0 otherwise. This prevents body parts of similar shape overlapping in 3D. This is important so that two parts do not try to explain the same data. More sophisticated penalty terms could be also used.

Minimize energy just on interfaces

Action plan

- Starting implementing it conceived algo
- Segmented fusion
  - Improve quality of mesh
  - Align globally in case of bigger leap between image
- Report

Q&A

- GPU code for merging vertices: size of variable?
  - Out of resource = overflow in variable
  - Memory allocation error = overflow in GPU memory
- Colors in Mesh?
- SP algo available