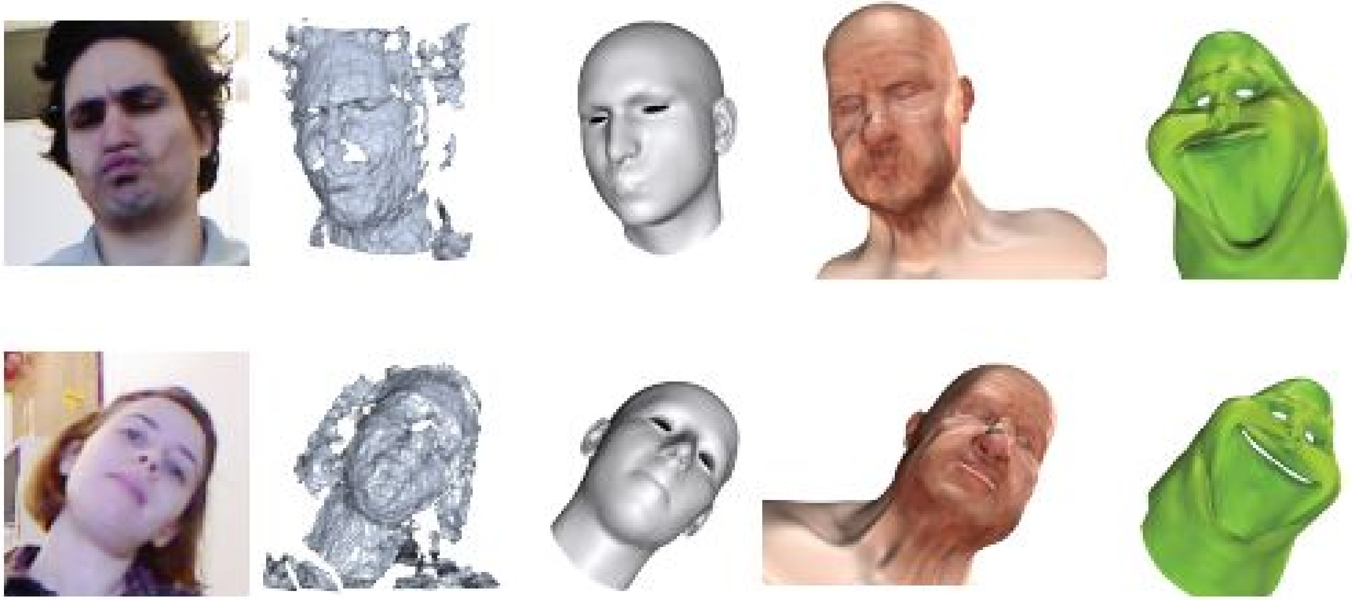
Realtime performance-based facial animation

Thibaut Weise, Sofien Bouaziz, Hao Li, Mark Pauly

Dibyendu Mondal 130050046
Anand Bhoraskar 130050025

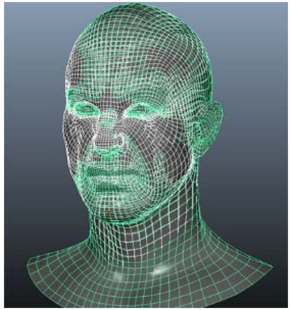
Aim



Introduction

- Create a low-cost facial animation system
- We use a non-intrusive, commercially available 3D sensor (Kinect)
- Markerless approach
- Face tracking algorithm that combines
  - Geometry registration
  - texture registration
  - Pre-recorded animation priors

PRE-REQUISITES



- Blendshape Representation (Morph Target Animation)
    - Neutral face is captured
    - Set of predefined expressions are captured (morph targets)
    - Animation frame is a blend of several morph targets
    - Represent facial expressions as weighted sum of blendshape meshes
    - Can be directly imported into commercial animation tools

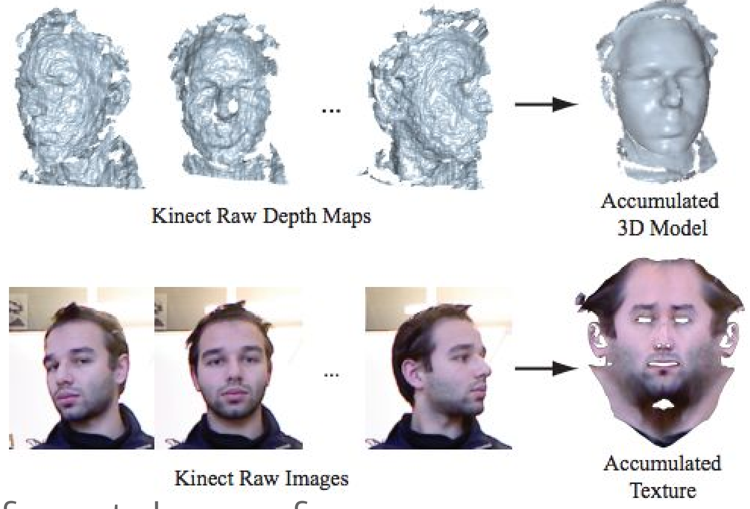
PRE-REQUISITES

- Acquisition Hardware
  - Kinect system is used
  - We capture a 2D color image and a 3D depth map
  - Not required to wear any physical markers or makeup

Facial expression model

- Data Capture
  - Record a predefined sequence of example expressions of the user
  - To prevent high noise levels, multiple scans over time are used
  - User is asked to perform slight head rotation while keeping the expression fixed
  - This has the additional benefit of alleviating reconstruction bias introduced by the spatially fixed infrared dot pattern

Facial expression model

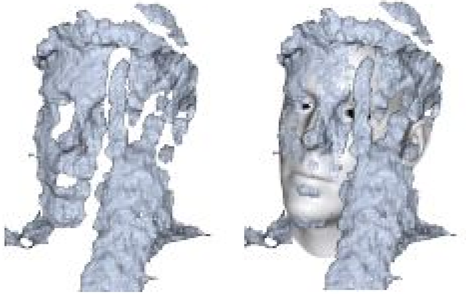


- Expression Reconstruction
    - Use morphable model to represent different human faces
    - A high quality template mesh roughly matching the geometry of the user’s face is obtained
    - Warp this template to each of the recorded expressions
    - To improve registration accuracy, we add texture constraints in the mouth and eye regions

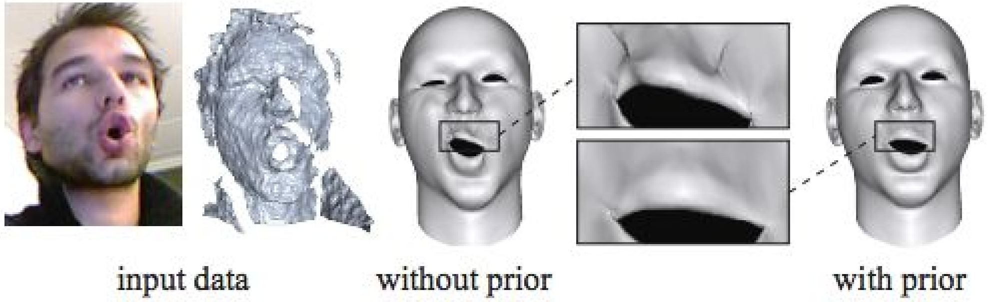
Facial expression model

- Blendshape Reconstruction
    - Represent dynamics of facial expressions using a generic blendshape rig based on Ekman's Facial Action coding System (FACS)
    - Employ example-based facial rigging:
        - Given data captured for all expressions and generic blendshape weights for all expressions
        - We reconstruct the set of user-specific blendshapes that best reproduce the example expressions

Realtime tracking



- Rigid Tracking
  - Use ICP(Iterative Closest Point) algorithm
  - Temporal filter with sliding window for handling high frequency flickering
- Non-rigid tracking
  - We use priors to make sure that the output is realistic



Statistical model

- MAP (Maximum a posteriori) estimation
  - $D = (G, I)$: input data at current frame i
  - G : depth map
  - I : color image
  - x : most probable blendshape weight
  - X_n : n previously constructed priors

$$x^* = \underset{x}{\operatorname{argmax}}\ p(x|D, X_n)$$

$$x^* = \underset{x}{\operatorname{argmax}}\ p(D|x, X_n)p(x, X_n)$$

$$x^* = \underset{x}{\operatorname{argmax}}\ \underset{\text{likelihood}}{p(D|x)}\,\underset{\text{prior}}{p(x, X_n)}$$

CONCLUSION

- High-quality performance-driven facial-animation in real time is possible
- Robust real time tracking achieved
- Combining animation priors with effective geometry and texture registration in a single MAP estimation is key
- Future scope:
    - Using real time speech analysis
    - Simulation of hair
    - Hand gestures

APPENDIX

Prior Distribution. To adequately capture the nonlinear structure of the dynamic expression space while still enabling realtime performance, we represent the prior term $p(\mathbf{x}, X_n)$ as a Mixtures of Probabilistic Principal Component Analyzers (MPPCA) [Tipping and Bishop 1999b]. Probabilistic principal component analysis (PPCA) (see [Tipping and Bishop 1999a]) defines the probability density function of some observed data $\mathbf{x} \in \mathbb{R}^s$ by assuming that $\mathbf{x}$ is a linear function of a latent variable $\mathbf{z} \in \mathbb{R}^t$ with $s > t$, i.e.,

$$\mathbf{x} = C\mathbf{z} + \mu + \epsilon, \tag{6}$$

where $\mathbf{z} \sim \mathcal{N}(0, I)$ is distributed according to a unit Gaussian, $C \in \mathbb{R}^{s \times t}$ is the matrix of principal components, μ is the mean vector, and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ is a Gaussian-distributed noise variable. The probability density of $\mathbf{x}$ can then be written as

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mu, CC^T + \sigma^2 I). \tag{7}$$

Using this formulation, we define the prior in Equation 5 as a weighted combination of K Gaussians

$$p(\mathbf{x}, X_n) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}, X_n|\mu_k, C_k C_k^T + \sigma_k^2 I). \tag{8}$$

with weights π_k. This representation can be interpreted as a reduced-dimension Gaussian mixture model that attempts to model the high-dimensional animation data with locally linear manifolds modeled with PPCA.

Likelihood Distribution. By assuming conditional independence, we can model the likelihood distribution in Equation 5 as the product $p(D|\mathbf{x}) = p(G|\mathbf{x})p(I|\mathbf{x})$. The two factors capture the alignment of the blendshape model with the acquired depth map and texture image, respectively. We represent the distribution of each likelihood term as a product of Gaussians, treating each vertex of the blendshape model independently.

Let V be the number of vertices in the template mesh and $B \in \mathbb{R}^{V \times m}$ the blendshape matrix. Each column of B defines a blendshape base mesh such that $B\mathbf{x}$ generates the blendshape representation of the current pose. We denote with $\mathbf{v}_i = (B\mathbf{x})_i$ the i-th vertex of the reconstructed mesh. The likelihood term $p(G|\mathbf{x})$ models a geometric registration in the spirit of non-rigid ICP by assuming a Gaussian distribution of the per-vertex point-plane distances

$$p(G|\mathbf{x}) = \prod_{i=1}^{V} \frac{1}{(2\pi\sigma_{\text{geo}}^2)^{\frac{3}{2}}} \exp(-\frac{||\mathbf{n}_i^T(\mathbf{v}_i - \mathbf{v}_i^*)||^2}{2\sigma_{\text{geo}}^2}), \tag{10}$$

where $\mathbf{n}_i$ is the surface normal at $\mathbf{v}_i$, and $\mathbf{v}_i^*$ is the corresponding closest point in the depth map G.

The likelihood term $p(I|\mathbf{x})$ models texture registration. Since we acquire the user's face texture when building the facial expression model (Figure 3), we can integrate model-based optical flow constraints [Decarlo and Metaxas 2000], by formulating the likelihood function using per-vertex Gaussian distributions as

$$p(I|\mathbf{x}) = \prod_{i=1}^{V} \frac{1}{2\pi\sigma_{\text{im}}^2} \exp(-\frac{||\nabla I_i^T(\mathbf{p}_i - \mathbf{p}_i^*)||^2}{2\sigma_{\text{im}}^2}), \tag{11}$$

where $\mathbf{p}_i$ is the projection of $\mathbf{v}_i$ into the image I, ∇I_i is the gradient of I at $\mathbf{p}_i$, and $\mathbf{p}_i^*$ is the corresponding point in the rendered texture image.

References

- http://dl.acm.org/citation.cfm?id=1964972
- http://dl.acm.org/citation.cfm?id=1272712

THANK YOU!