

Particle Filter on GPUs for Multiple Object Tracking in HCI Applications

Antonio S. Montemayor, Juan José Pantrigo, Ángel Sánchez
ESCET - Universidad Rey Juan Carlos (Madrid, SPAIN)
{antonio.sanz, juanjo.pantrigo, angel.sanchez}@urjc.es

Felipe Fernández
FI - UPM (Madrid, SPAIN)
felipe.fernandez@es.bosch.com

1 Introduction

Human-Computer Interaction (HCI) is evolving towards non-contact devices, using perceptual and multimodal user interfaces. Computer Vision is very useful for achieving this goal, providing new ways of interaction. Human body tracking by monocular vision is an important task for the development of such systems.

Recent research in human motion analysis makes use of the Particle Filter (PF) framework. PF algorithm enables the modeling of a stochastic process with an arbitrary probability density function (pdf), by approximating it numerically with a set of samples called particles in a state-space process [Arulampalam et al. 2002]. The main aim of particle filtering is the accurate tracking of a variable of interest as it evolves over time, such as kinematic information of objects in video sequences.

In this work, we propose some improvements to a previous particle filter framework [Montemayor et al. 2004]. In particular, we have implemented a real-time Joint Probability Distribution Particle Filtering for multiple object tracking in monocular video sequences. This computationally demanding task is processed on the graphics processing unit (GPU) in a stream programming model.

2 Method Overview

Figure 1 outlines an iteration of the proposed particle filter algorithm that tracks 3 regions of interest (ROI). Note that in the GPU/CPU implementation scheme, the GPU side is shaded. The performance of the filter has been tested on a video sequence in which both hands and head of a user are tracked. The video frame at time t is loaded into video memory as a texture data (Tex0) and a skin detection fragment shader is applied to it in order to obtain a measure (Figure 1.a). Although this is not the core of the method, a pixel (R, G, B) is classified as skin if [Peer et al. 2003]:

$$(R > 45) \ \& \ (G > 40) \ \& \ (B > 20) \ \& \ (R > G) \ \& \ (R > B) \ \& \ (\max(R, G, B) - \min(R, G, B) > 15) \ \& \ (|R - G| > 15)$$

Then, N samples consisting of 3 square regions are taken from the binary image (Figure 1.b). So each particle j carries information of (x_i, y_i) subwindow coordinates ($i = 1..3$) and a global weight π_t^j for this 3 ROIs configuration. In the first iteration this sampling is randomly generated from a uniform pdf.

An entire RGB texture (Tex1) is filled by arranging and loading each subwindow in separate channels as Figure 1.c shows. This fact is crucial for performance reasons. Particle weight computation is based on a template matching approach although we have previously sampled the original image at discrete locations instead of performing an exhaustive search. In order to improve this stage, a simple fragment program is created to carry out an effective hardware accelerated evaluation. This shader calculates the average of the RGB components per pixel. Then, and as we chose square subregions for each measurement, a mipmap reduction is accomplished to get an estimation of the particle weights π_t^j at each pixel position. In order to get an estimation refinement, we perform a resampling stage (see Figure 1.d.1) evaluating previous particle weights and

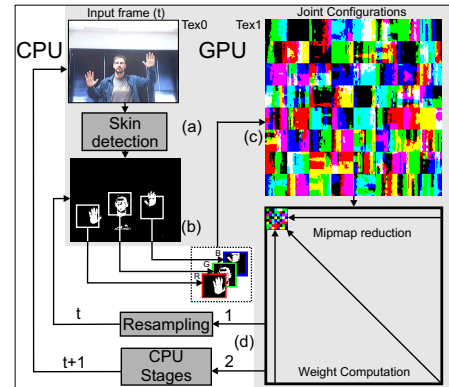


Figure 1: Hybrid GPU/CPU particle filter scheme.

concentrating particles around the most probable states, discarding those with lower weights.

Next, we get the second pass rendering results to proceed with the following stages of the particle filter which are carried in the CPU (shown in Figure 1.d.2), that are more deeply explained in [Montemayor et al. 2004]. The particle x_t^{max} with the maximum weight is selected as best candidate for the state of the system in the iteration. In other words, the 3 best subwindow locations stored in the particle with maximum weight describe the position of hands and head.

3 Conclusion

A real-time joint probability distribution particle filter that exploits the intrinsic parallelism of the GPU architecture has been implemented. It successfully operates at 49 fps in a 3.2 GHz Pentium 4, 1 GB RAM equipped with a commodity NV34 graphics card for 256x256 video resolutions. This framework is applied to a human-computer interaction problem in which both hands and head of a user are tracked. This tracking task can be further refined in order to include a gesture recognition stage for an intelligent user interface.

References

- ARULAMPALAM, S., MASKELL, S., GORDON, N., AND CLAPP, T. 2002. A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing* 50, 2, 174–188.
- MONTMAYOR, A., PANTRIGO, J., SÁNCHEZ, A., AND FERNÁNDEZ, F. 2004. Particle filter on GPUs for real-time tracking. In *Proc. of ACM SIGGRAPH 2004*.
- PEER, P., KOVAC, J., AND SOLINA, F. 2003. Human skin colour clustering for face detection. In *Proc. of the International Conference on Computer as a Tool EUROCON*.