Discussion on the design of Vision Sub Architecture (VSA)

February 8, 2008

1 Introduction

The goal of this discussion is to pull together people who work on vision across 4 research sites (BHAM, UOL, TUD, KTH) to come to soft decision on the design of the VSA for the final year review. The discussion proceeds by a representative from each site explainging the existing vision components that have been developed and will be integrated for demonstation. The following is the list of these components, short summary of their functionalities and pointers to appropriate data structures in the idl files.

We aim to determine whether where are any overlaps in vision fuctionalities and resolve any disagreement on common components that may be used in both of scenario demontrations.

2 Overview of vision-related components

2.1 Components maintained by BHAM

The following are the the list of component inside the VSA. MC means a "managed component", and UMC an "unmanaged component".

- Task manager
- VisualStateGenerator (MC)
- Trigger (UMC)
- Segmentor(MC)
- Sift (MC)
- Saliency (MC)
- Color Classifier (MC)
- Shape Classsifer (MC)

- Perceptual Grouper (MC)
- Surface Detector (MC)
- Comparator (MC)
- Binding Monitor (MC)
- Video server (UMC) supports various drivers (e.g., firewire, v4l).
- Change detector (UMC)
- ObjectTracker (UMC) pulls images from VS, tracks objects using kalman filtering based on color and edges. Produces SceneObject and ROI in WM.

The exchange of information between processes from different components are by encapsulating the information in a data structure (written in italic). In most cases, this data is deposited into the visual working memory (VWM), which triggers processing in other components. but some of them are passed around directly (e.g., pulling or requesting) between components. The following sections describes some of the main processes that perform this task in the current system.

2.1.1 General Vision Processing

General vision process starts with **ChangeDetector** pulling an *ImageFrame* from **VideoServer**, processing the image to detect a scene change and posting a *SceneChanged* in VWM. Triggered by the *SceneChanged* added in VWM **Segmentor** requests an *ImageFrame* from **VideoServer**, extracts regions of interest and writes the extracted *ROIs* and their corresponding *SceneObject* to VWM. Both added *ROIs* and *SceneObject* may then trigger other visual processes that are listening for these events.

2.1.2 Processes related to Visual Planning

Triggered by a question like "what do you see?" from ComSysSA, visual planner determines a sequence of operators to perform.

For testing, Trigger in VSA can monitor ROIs in VWM and write a PlanningProcessRequest (Domain,SA) in the WM of PlanningSA (e.g., (ContainsColor ?x red)). This alternatively alerts PlanningStateGenerator in VSA. After running MB's planner, Actions are written in VWM (e.g., run colour, shape, sift, etc.).

2.1.3 Vision Processes in ManipulationSA

Vision processes in ManipulationSA make use of assumptions about the top view to search for rectangles or ellipses. SurfaceDetector groups detect and group edges, look for closure to detect rectangles and ellipses. Some constraints (e.g., on sizes, closeness to center) are used to choose reliable rectangles.

2.2 Components maintained by UOL

UOL has developed the following components for learning and recognizing object properties: Segmentor, FeatureExtractor (FE) that performs FE on a ROI of SceneObject, Matlab-based Learner/Recogniser, which produces the color and shape types for each ROI. Triggered by a ROI/SceneObject being added to VWM, BindingMonitor produces a proxy in BindingSA. The task of Comparator is to coordinate matching between color/shape strings from Comsys to labels used in Learner/Recogniser.

2.3 Components in Explorer (KTH, TUD)

At KTH, only NavSA is adapted to CAST. Many vision components are specific to explorer scenarios (to support large-scale navigation, to follow a person, to go to a person) and may not be required in Playmate. Some vision functionalities include ObjectSearch, ViewPlanning, Database, Translation.

Vision functionalities in Explorer are designed to support functionaloties of NavSA which has the following components: **Slam**, **BuildNavGraph**, **NavControl** (interacting with vision and Laser), **PlaceRecognition** (interacting with framegrabber and laser server) **PeopleRecogniser** (interacting with framegrabber and laser server).

The general procession cycle starts with *ObjectSearchCommand* is received from PlannerSA. **Translation** translates *ObjectSearchCommand* into *ViewPlanningCommand*. **ViewPlanning** receives *ViewPlanningCommand* and triggers processes in NavSA (e.g., to build NavGraph (1-2meters between graph nodes), use it to go to a location node in the graph, and to turn into a certain direction). After navigation, NavSA writes *VisualSearchCommand* (containing info like find(Book), current camera angle, etc.) which triggers **ObjectSearch** to grab frames, talk to PTZServer to orient camera, to search for the book. When the book is found, it sends *Acknowledgement* and the found object can be inserted into NavMap by **BuildNavGraph**.

3 Some decisions to be made

1) Some vision components developed at KTH or TUD that are outside VisionSA but might be useful for others may be moved to VisionSA? E.g., PeopleDetector may be moved to VisionSA? (Gyuri has left the project, so maybe Niko can do this?)

2) Currently not all sites use the VideoServer made avilable in SVN. Can PeopleTracker (TUD) and PanTiltStereo (GJ) components be changed to use BHAM's VideoServer?