Chapter 7

APPLICATIONS OF LINEAR GENETIC PROGRAMMING FOR OBJECT RECOGNITION

7.1 Introduction

This chapter is a logical continuation of chapter 6 and presents results of applying the methodology described there to real-world computer vision and pattern recognition problems. In particular, the configurations verified here include basic, single-population evolutionary feature programming (EFP), and selected variants of coevolutionary feature programming (CFP) working on different decomposition levels.

To provide experimental evidence for the generality of the proposed approach, we verify it on two different tasks. First of them is the recognition of common household objects, a popular benchmark used in computer vision community. It concerns the visible part of the electromagnetic spectrum and relates to so-called passive sensing, as usually no active dedicated source of light is required to acquire the images. On the contrary, the second considered application concerns the non-visual modality of radar imaging and represents active sensing, as the source of radiation (radar wave transmitter) is required. Therefore, the problems considered are entirely different; the only features they have in common are (a) recognition of 3D objects from different viewpoints, and (b) using mid-size one-channel raster images.

7.2 Technical Implementation

To provide an experimental testbed we developed a software environment named CVGP (Computer Vision by Genetic Programming). CVGP, written in Java and C, is a universal platform for experimenting with explicit feature construction in both machine learning and computer vision. To conform to the existing standards and benefit from the ready-to-use background knowledge, CVGP integrates several existing libraries:

- Soft-computing libraries written in Java:
  - machine learning library WEKA [127],
  - evolutionary computation library ECJ [70].
- Image processing and computer vision libraries (C and machine code):
  - Intel Image Processing Library (IPL) [47],
  - Open Computer Vision Library (OpenCV) [88],

Figure 7.1 presents the overall software architecture of the system. Java Native Interface (JNI) has been used to integrate modules and libraries written in Java with those written in C. Thanks to this choice of components, the most time-consuming evaluation of Feature Extraction Procedures (FEP) is efficiently carried out in well optimized libraries written in C and machine code, whereas the less computationally demanding ML and EC computation takes place in Java. The IPL and OpenCV libraries function as a repository of background knowledge. Though originally designed to serve explicit feature construction in CV, CVGP may also be applied to ML problems; in such a case, WEKA and ECJ are sufficient to run an experiment. On the other hand, CVGP may be easily combined with other libraries to use background knowledge and input representation relating to other domains (sound, video, etc.).