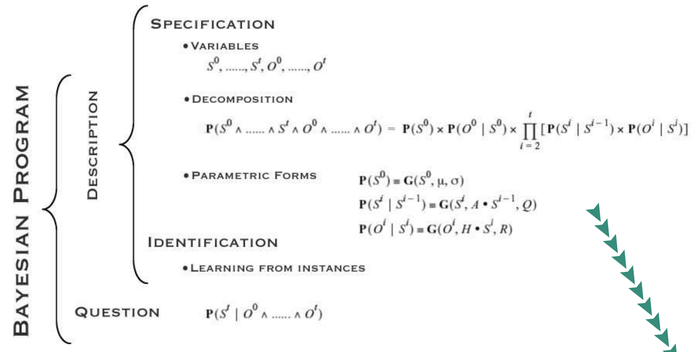
Bayesian Programming formalism and ProBT API

J-M Ahuactzin, E. Mazer & K. Mekhnacha
ProBAYES

P. Bessière
CNRS
Pierre.Bessiere@imag.fr

Industry

www.probayes.com



Inference Algorithms: Junction Tree Algorithm, Successive Restriction Algorithm (patented), MCMC, Monte Carlo Simultaneous Estimation and Maximization (Patented), Multi Rsolution Binary Tree (patented)

Learning Algorithms: Maximum Likelihood, Conjugate laws, Expectation Maximization MSWT, K2, MSWT-EM, K2-EM

Products:

- ProBT®
- ProBT- Excel
- Bayesian Occupancy Filter

Applications:

Bank-Finance:

- Operational Risk
- Fraud Detection
- Stock picking

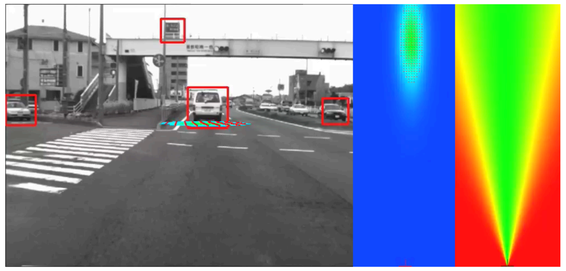
Industry:

- Automotive Industry
- Preventive Maintenance

Services:

- Logistic
- Cost and price evaluation
- Intelligent Marketing
- Energy

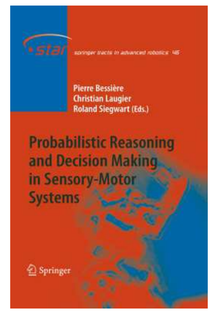
Defense and Security




```
/*=============================================================================
* Product     :
* File        : diceFilter.cpp
* Author      : Juan-Manuel Ahuactzin
* Creation    : 2008-Oct-02 16:16
*
*=============================================================================
*               (c) Copyright 2008, Probayes SAS,
*                   all rights reserved
*=============================================================================
*
*------------------------- Description ---------------------------------------
* This program computes the more probable number of dice given the
* sequence of number of points corresponding to throwing the dice t
* times.
*-----------------------------------------------------------------------------
*/

#include <pl.h>

using namespace std;

#define SQRT_DIV35_12 1.7078251


// User function for computing the mean
void f_mean(plValues &mean, const plValues &n) {
 mean[0] = 7*n[0]/2;
}

// User function for computing the standard deviation
void f_std(plValues &std, const plValues &n) {
 std[0] = SQRT_DIV35_12*sqrt(double(n[0]));
}

int main() {
/*****************************************************************************
 VARIABLES SPECIFICATION
*****************************************************************************/
 unsigned int max_n;
 cout<<"Give me the maximum number of dice: ";
 cin>>max_n;

 plIntegerType die_number(1,max_n);   // Type for the maximum number
                                      // of dice [1,2,...,max_n]
 plIntegerType dice_sum(1,max_n*6);   // Type for the sum [2,3,...,n*6]
 plSymbol points("Addition",dice_sum); // Variable space for the addition
 plSymbol number("n",die_number);     // Variable space for the
                                      // selected number of dice

 plValues result(number^points);      // Values storing the # of dice
                                      // and the addition
/*****************************************************************************
 PARAMETRIC FORM SPECIFICATION
*****************************************************************************/
 // P(number) = uniform(number)
 plUniform p_number(number);

 // Create the external functions to compute the mean and the
 // standard deviation
 plExternalFunction f_mu(number,f_mean);
 plExternalFunction f_sigma(number,f_std);

 // P(points | number) = CndBellShape(points,f_mu(number),f_sigma(number))
 plCndBellShape p_addition(points,number,f_mu,f_sigma);

/*****************************************************************************
 DECOMPOSITION
*****************************************************************************/
 // P(points number) = P(points | number) P(number)
 plJointDistribution dice_jd(number^points,p_number*p_addition);
 dice_jd.draw_graph("addDice3.fig");

/*****************************************************************************
 PROGRAM QUESTION
*****************************************************************************/
 plCndDistribution cnd_question;
 plDistribution question;
 int v;

 // Get P(number | points)
 dice_jd.ask(cnd_question,number,points);

 unsigned int t=0;

 cout<<"Give me the number of throws? ";
 cin>>t;

 do {
  t--;
  cout<<"Give me the number of points: ";
  cin>>v;
  result[points] = v;

  // Get P(number | points = v)
  cnd_question.instantiate(question,result);
  plDistribution compiled_question;

  question.compile(compiled_question);
  cout << "The distribution is:\n" << compiled_question << endl;
  compiled_question.best(result);

  cout<<"\nThe more probable is: "<<result[number]<<"\n";

  cnd_question.replace(number, compiled_question);
 } while (t > 0);

 return 0;
}
```

Research

www.bayesian-programming.org



[1]

Robotics:

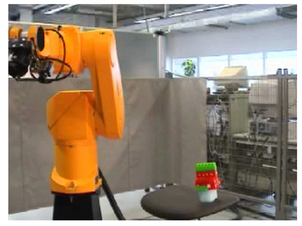
- Bayesian Robot Programming [2]
- The CyCab: Bayesian Navigation on Sensory-Motor Trajectories [3]
- The Bayesian Occupancy Filter [4]
- Tological SLAM
- Bayesian Maps: Probabilistic an Hierarchical Models for Mobile Robots Navigation
- Bayesian Approach to Action Selection and Attention Focusing
- Bayesian Control of a Robotic Arm

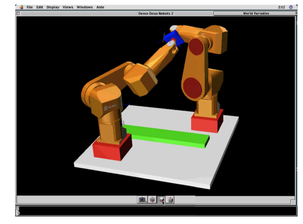
Industrial applications:

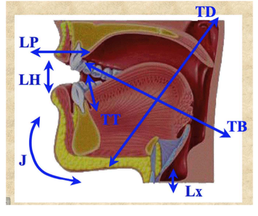
- BCAD: A bayesian CAD System for Geometric Problems Specification and Resolution [5]
- 3D Human Hip Volume Reconstruction with Incomplete Multimodal Medical Images: Application to Computer-Assisted surgery
- Playing Train Video Game Avatar [6]

Cognitive Modeling:

- Bayesian Modeling of Visuo-Vstibular Interactions [7]
- Bayesian Modeling of Perception of Structure from Motion [8]
- Building a Talking Baby Robot: Study of Early Speech Acquisition [9]
- Bayesian Model of the Superior Colliculus [10]







We are offering:

A free version of ProBT® for academic an research purposes

We are looking for:

Collaborations and inference or learning algorithms to implement in ProBT

[1] - Probabilistic Reasoning and Decision Making in Sensory-Motor Systems, Bessière, P., Laugiare, C. & Siegwart, R., SPINGER (2008) [2] - Bayesian Robot Programming, Lebeltel O., Bessière P., Diard J., Mazer E., Autonomous Robots 16, 1 (2004) 49--79 [3] - The CyCab: a car-like robot navigating autonomously and safely among pedestrians, Pradalier C., Hermosillo J., Koike C., Braillon C., Bessière P., Laugier C., Robotics and Autonomous Systems 50, 1 (2005) 51-68 [4] - Bayesian Occupancy Filtering for Multitarget Tracking: an Automotive Application, Coué C., Pradalier C., Laugier C., Fraichard T., Bessière P., Int. Journal of Robotics Research 25, 1 (2006) 19--30 [5] - The Design and Implementation of a Bayesian CAD Modeler for Robotic Applications, Mekhnacha K., Mazer E., Bessiere P., Advanced Robotics 15 (2001) N.1, pp 45-70 [6] - Teaching Bayesian Behaviours to Video Game Characters, Le Hy R., Arrigoni A., Bessière P., Lebeltel O., Robotics and Autonomous Systems 47 (2004) 177--185 [7] - Bayesian Processing of Vestibular Information, Laurens, J. & Droulez, J., Biological Cybernetics (2007) Vol96(4), pp. 398-404 [8] - A unified probabilistic model of the perception of three-dimensional structure from optic flow, Colas F., Droulez J., Wexler M., Bessière P., Biological Cybernetics (2008) 132--154 [9] - Building a talking baby robot A contribution to the study of speech acquisition and evolution, Serkhane J., Schwartz J.-L., Bessière P., Interaction Studies 6, 2 (2005) 253–286 [10] - Bayesian models of eye movement selection with retinotopic maps, Colas F., Flacher F., Tanner T., Bessière P., Girard B., Biological Cybernetics (2009)