//Utterances for Autowow Oct2007

//--

Here is a short example of how the English version of a 'urlaut', a one dimensional string of letters, is evolved based on a fitness function that favors certain vowel sequences. The rules are different for other languages (German, French, Spanish) and not included here. The rules for languages the system can not simulate (non-European languages) are not yet addressed in the project.
The look-up table that maps gps location to a language of choice is not included in this example.
The function that receives the battery level (and used to modify the duration of the utterance) from the telescope is not included here.
The setting and modification of prosody (F0, intonation) are a separate program not included here.

```
//--------------------------------------------------------------------------------------------------------------------------------------------------
//globals
short alphabetsize = 26;
char specials[] = {'h', 'w'};
char specialvowels[] = {'a', 'o', 'u'};
bool verbose = 1;
//--------------------------------------------------------------------------------------------------------------------------------------------------
void main()
{
srand((unsigned)time(0));
for(int ii=1; ii<alphabetsize; ii++)
GARandomSeed((unsigned int) ii);
GAStringAlleleSet alleles;
for(short i=0; i<alphabetsize; i++)
alleles.add('a'+ i);
GAStringGenome genome(alphabetsize, alleles, objective);
genome.initializer(alphabet_initializer);
genome.mutator(GAStringSwapMutator);
GASteadyStateGA ga(genome);
ga.populationSize(100);
ga.pCrossover(0.95); //0.9
ga.pMutation(0.1); //0.01
ga.nGenerations(100);
ga.crossover(GAStringPartialMatchCrossover);
ga.initialize();
//prior to evolving
if(verbose)
{ for(i=0; i<ga.population().size(); i++)
cout << ga.population().individual(i) << endl;
}
//evolve
while(!ga.done()) ga.step();
//after evolving
if(verbose)
{ for(i=0; i<ga.population().size(); i++)
cout << ga.population().individual(i) << endl;
}
if(verbose)
{ cout << "\nbest individual: ";
genome = ga.statistics().bestIndividual();
cout << "\n" << genome;
}
//parse result
cout << "\n\nparsed string: ";
string finalstring = parse_result(ga, alphabetsize, specialvowels);
cout << "\n" << finalstring << "\n";
}
//------------------------------------------------------------------------------------------------------------------------------
```