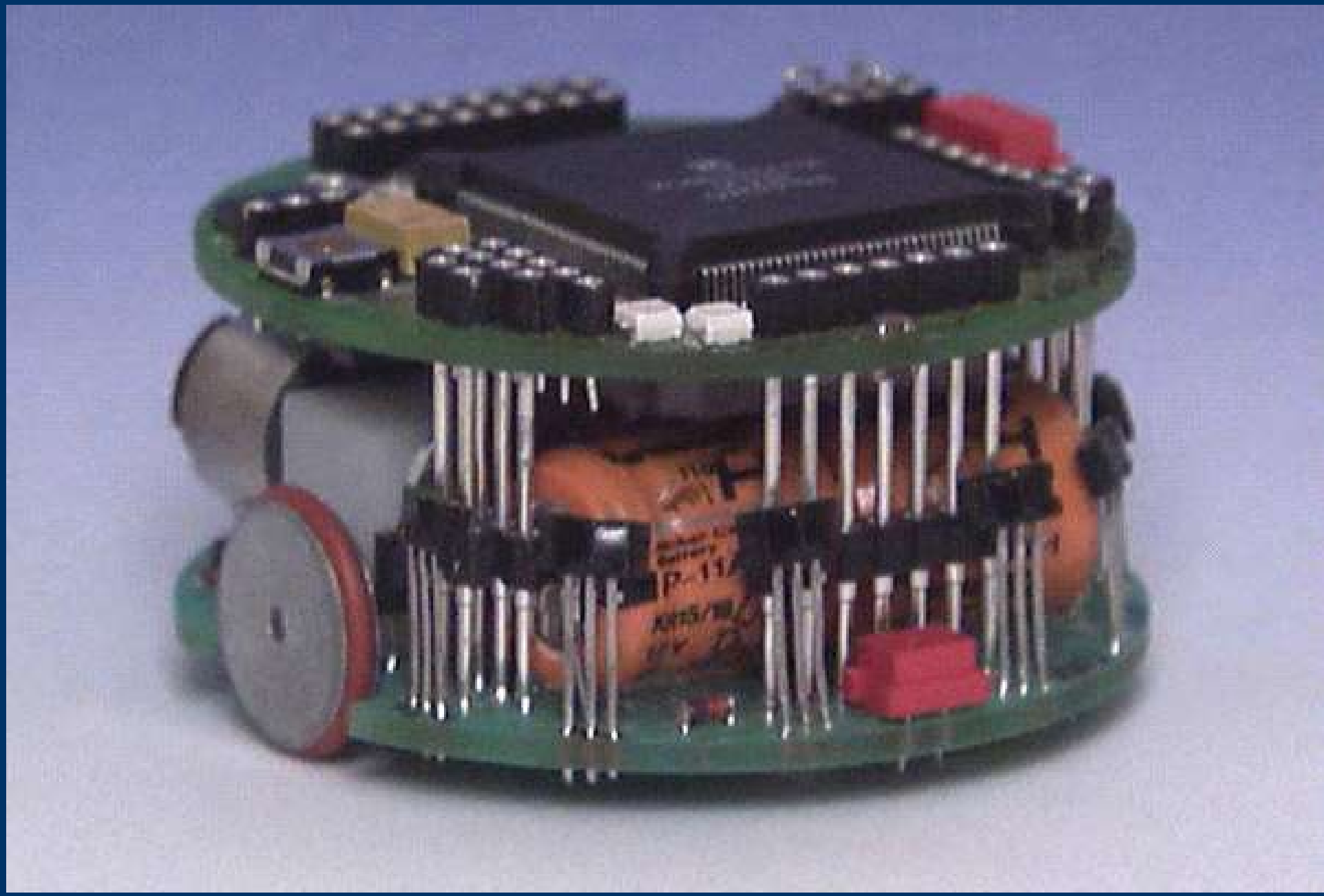
Mid-Semester Report:

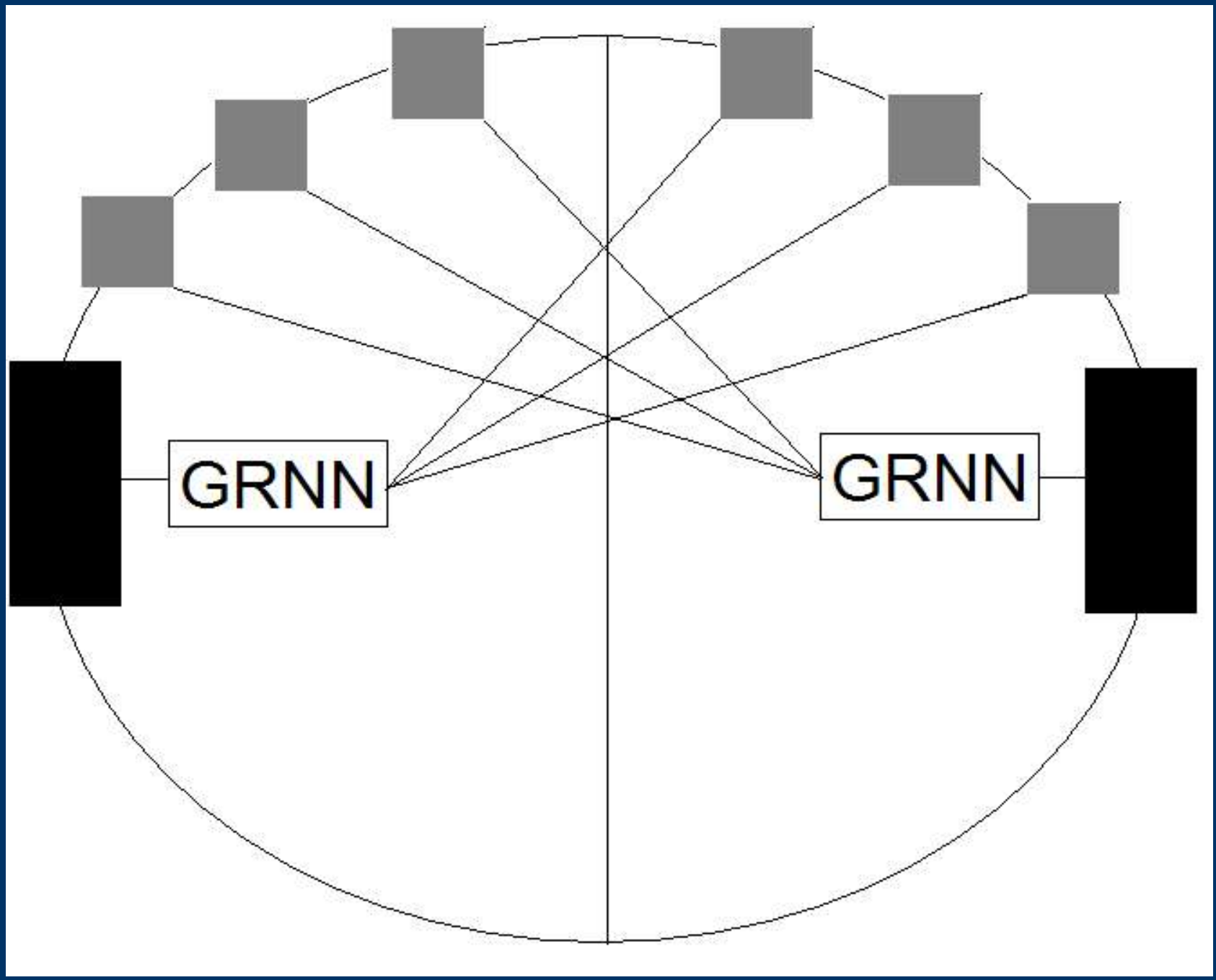
Dozier's Amazing Evolutionary Khepera

As presented by Travis Hite



Khepera Design Aspects

- 55 mm in diameter
- 33 mm tall
- 68331 Motorola processor
- 8 infra-red sensors
- Max speed 127 mp (motor pulses)
  - Roughly one meter per second


GRNN
GRNN

The Neural Network

- Hemispheres
- 3 sensors for proximity detection
- 1 wheel per 3 sensors
- GRNN

The Stage

- 30x30cm
- White cushioned cardboard
- Wall proximity detection and avoidance

The Sensors

- Each sensor connects to GRNN
- GRNN holds three responses
  - {1023,0,0} Front
  - {0,1023,0} Mid-Front
  - {0,0,1023} Mid-Back

The Inidividuals

Four values
First three weights range from -30 to 30
- Effects wheel speed
- Each corresponds to a specific sensor

Final rate ranges from 0 to 512
- Effects overall motor speed
- “Sigma value”

Example individual: {-20,4,18,423}

The IEC

1) Random population is generated
2) Each individual controls the robot for 4 seconds*
3) User interfaces with robot
4) 2 individual tournament
5) Least hits wins
6) Construct difference rules
7) Evolve individuals

*unless aborted

The Difference Rules

Basic distance comparison between two individuals
Four integer values {-1,0,1}
One interger value {-1,1}
Example:

p1 = {-14,-25,-17,136}
p2 = {-20,-25,-22,386}
v = < 1, 0, 1, -1, 1>

Evolution Rules

Candidate was aborted previously – sign mutation

p1={-14,28,-12,489}

result={-14,-28,-12,489}

No feasible result – uniform crossover

p1={-12,-18,-24,582}

p2={-28,-4,16,212}

result={-12,-4,-24,212}

Both aborted – feasible candidate from population receives uniform-bounded mutation twice, then parent is mutated as well .

p1={-14,-28,-12,489}

result={-21,-19,-4,456}

One feasible parent – uniform-bounded mutation

The MEC

Khepera stops moving to “meditate”
Continues tournament, generates distance vector
Uses distance vectors generated during IEC
Evolution continues as previously mentioned

Problems

User error rate exponential
Sigma value hard to control
Interface could be better

The Interface

Java webcam interface:

- Too much delay
- Read, buffer, stream, receive

The Weights

Adjusting for user error

- One idea: ELO Rating

$$R'_a = R_a + K(S_a - E_a) \qquad E_a = \frac{1}{1+10^{\frac{R_a - R_b}{400}}}$$

Bayesian Modifiers

Originated in expert systems

- If-Then data complex

Based on simple probability between a hypothetical situation and an event

A basic implementation would look like this:

$$p(A|B) = \frac{p(B|A) \times p(A)}{p(B)}$$

Modified for Our Purposes

If we only evaluate for one function of our weights depending upon the winning value of the distance value, p(A) can effectively be made 100%

Also, if we flip all the values when the winning distance is -1, we can effectively make p(B) 100%

Example:

<1,0,-1,1,-1> becomes <-1,0,1,-1,1>

Applying for our purposes

Mutation rate can be adjusted on a sliding scale based on user preference.

For instance, take a standard mutation rate for the first weight being {-12...12}, and we receive a -1 during 25/35 evaluations

We can adjust the mutation rate accordingly:

.75*(25/35)*-1 = -0.536

.75 applied to keep mutation rate from reaching 100% and completely biasing our search

New mutation rate: {-18...6}

Problems

Further amplification of user-related false negative results
Further amplification of sigma value problems

Questions or comments?