Humboldt-Universität zu Berlin
Mathematische-Naturwissenschaftliche Fakultät
Institut für Informatik

Self-Healing in Swarm Robotic Systems

seminar work

by Vincent Becker and Mark Schatz

Contents

1 Abstract

In our seminar work we are dealing with a way to give swarm robotic systems some more robustness. There are approaches to optimize the algorithms for their different tasks. The one we are coping with is a more general one. Self-healing can be adopted for a wider field of missions. Nevertheless we are coping with swarm taxis. Based on the paper "An Immune-Inspired Swarm Aggregation Algorithm for Self-Healing Swarm Robotic Systems" by Timmis, Ismail, Bjerknes and Winfield we are describing solutions to robot failures.

2 Introduction

2.1 Swarm Robotic Systems

Swarm behaviour is the decentralised coordination of many self-propelled entities. They got simple rules for their local interactions and communication which lead to an emergent behaviour. When talking about swarm robotic systems we got a number of at least 5 robots with simple algorithms which only got their local knowledge. Those systems need to fulfil their task in a robust, flexible and scalable way.

2.2 Swarm Taxis and Self-Healing

One very simple task for robot swarms is moving towards a beacon, called swarm taxis. We will focus on swarms build to achieve this task. Timmis and Co worked with a swarm of 10 e-Puck robots.

2.2.1 A Simple Task - Swarm Taxis

E-puck robots are equipped with an omnidirectional light sensor, 3 proximity sensors, a short range, omnidirectional radio device and some LEDs. This is one possible set of tools for swarm taxis robots which need to go towards a light emitting beacon while avoiding obstacles on their way.

There could be confusion about the terms self-healing and self-repairing.
So on one hand we refer to self-repair when talking about reaching the target in terms of the swarm despite different robot failures. However there is no repair of certain robots. It's just the remaining swarm achieving it's aim.
On the other hand we refer to self-healing when talking about reaching the target in terms of a single robot despite partial failures. Here we focus on sharing energy between robots to get up a robot which was low on energy.

In their setting Timmis and Co identified three failure modes for a certain robot:

total failure the robot completely stops and just behaves like an obstacle - self-repair

failure of infrared sensors the robot becomes a moving obstacle and get lost for the swarm - self-repair

failure of motors only the robot stops moving (e.g. due to low energy) but still sends signals - ...

The last mentioned failure mode is the most fatal one. In this case the failed robot will anchor the whole swarm by its position. With just one failed robot the swarm still might move on with some delay. However the more motors of different robots fail the more likely it is that the swarm can't move on and is finally anchored by those failed robots.

As mentioned before motor failure is often induced by low energy. So the idea is to charge robots which are on low energy which could be one application of self-healing. In the following we are describing the swarm taxis algorithms first, followed by simple self-healing algorithms before we will explain a more complex one in our next chapter.

3 Swarm Taxis Algorithms

3.0.1 α-Algorithm

α-threshold
limit of minimum neighbours
when less, reconnect
180°-turn
when reconnected, random heading

3.0.2 β-Algorithm

β-threshold
limit of minimum neighbours still having connection to a lost robot
when less, reconnect
same as in α-Algorithm

3.0.3 ω-Algorithm

ω-threshold
limit maximum of ticks since last avoidance behaviour
when more, reconnect
turn to estimated swarm centre
when reconnected, random heading

to calculate swarm centre replaced wireless connection by timer and ring of IR-emitters

swarm even more connected than with β-Algorithm

4 Self-Healing Algorithms

We discussed the problem of anchoring and tried to solve this problem with the approach of swarm taxis algorithms. The swarm beacon taxis, tried to tell us the problem of, what can happen if we don't solve the problem of dead robots, who still sending signals.

4.1 Single-Nearest-Charger Algorithm

One approach is the idea of a Single-Nearest Charger. The idea behind this approach is that robots near the failed robot, send informations about their level of energy and distance away from the failed robot.
We want to conclude, how robots will act, if one robot had failed. So if one robot has some malfunction and cannot move any further, that would not be a problem for our case, because the robots would handle this failed robot as an object. Our Problem here is, that the robot can still send signals to other robots. These signals are position data and data about the current state. All robots work with the same algorithm, so everyone is trying to stay near to him, anchoring occurs.

(insert picture here, to demonstrate, simulation)
The Hypotheses behind this, is built as followed:

> *The use of single nearest charger mechanism (M2) for swarm beacon taxis does not improve the ability of the robots in the systems to reach the beacon when compared with M1 when more than two faulty robots are introduced to the systems.*

In contrast of the form of 'central sharing', robots must be able to distribute the collective energy resources owned by the group member. This means that we need a simple solution to share the energy between the robots. Here calls the single nearest charger algorithm. We assume the simplest rules, which are the position of the robot in the environment and the limit of energy transfer. Robots can donor or receive energy from only one robot at the same time. We say that the nearest robot acts as the donor if one failed robot is sending signals.

The issue about this idea, is the fact, that we can not control the amount of energy that is transferred from donor to receiver or vice versa. So the problem will be, that one failed robot get half of the energy of one donor, the donor now needs energy and a donor for himself. That brings another anchoring effect.

Due to problems with the recharge of the failing robots by docking on them (3 robots, for 1 failed robot) we need a solution, to not interfere with the other helpers so a constant energy flow can happen.

Algorithm 1 Single-Nearest-Charger Algorithm

```
procedure DONOR ENERGY
    Deployment: robots are deployed in the environment
    repeat
        Random movement of the robot in the environment Signal propagation: Faulty robots will emit faulty signal Rescue: One of the healthy robots with the nearest distance (earliest arriving robot) will perform protection and rescue Repair: Sharing of energy between faulty and healthy robots according to algorithm 2
    until forever
```

Algorithm 2 Algorithm for containment and repair for single nearest charger algorithm

```
procedure
    begin
    Evaluate pos_self(t)
    Send pos_self(t) to peers
    Receive egy_peer(t − x) and pos_self(t − x) from peers
    for all egy_peer(t − x) do
        Evaluate egy_peer(t − x)
        if egy_peer(t − x) < egy_min then
            Evaluate egy_peer(t − x)
            Sort pos_peer(t − x) in ascending order
            Move to nearest pos_peer(t − x)
        else
            Do not move to pos_peer(t − x)
        end
    end
```

4.1.1 Shared-Nearest-Charger Algorithm

An approach to avoid the failure, that one robot donor to one robot and both will fail. Is the idea of Shared-Charging. The failed robot will send out information, about his location, in detail the distance between it and the failed robot and the actual energy level. If the level is ok and one robot is closer to the failed robot than the others