World Academy of Science, Engineering and Technology
International Journal of Mechanical and Materials Engineering
Vol:13, No:9, 2019

An Application of Path Planning Algorithms for Autonomous Inspection of Buried Pipes with Swarm Robots

Richard Molyneux, Christopher Parrott, Kirill Horoshenkov

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

***Abstract*—**This paper aims to demonstrate how various algorithms can be implemented within swarms of autonomous robots to provide continuous inspection within underground pipeline networks. Current methods of fault detection within pipes are costly, time consuming and inefficient. As such, solutions tend toward a more reactive approach, repairing faults, as opposed to proactively seeking leaks and blockages. The paper presents an efficient inspection method, showing that autonomous swarm robotics is a viable way of monitoring underground infrastructure. Tailored adaptations of various Vehicle Routing Problems (VRP) and path-planning algorithms provide a customised inspection procedure for complicated networks of underground pipes. The performance of multiple algorithms is compared to determine their effectiveness and feasibility. Notable inspirations come from ant colonies and *stigmergy*, graph theory, the k-Chinese Postman Problem (k-CPP) and traffic theory. Unlike most swarm behaviours which rely on fast communication between agents, underground pipe networks are a highly challenging communication environment with extremely limited communication ranges. This is due to the extreme variability in the pipe conditions and relatively high attenuation of acoustic and radio waves with which robots would usually communicate. This paper illustrates how to optimise the inspection process and how to increase the frequency with which the robots pass each other, without compromising the routes they are able to take to cover the whole network.

***Keywords*—**Autonomous inspection, buried pipes, stigmergy, swarm intelligence, vehicle routing problem.

I. Introduction

WATER networks cover the UK and the conditions of the buried pipelines is ever-decreasing due to poor inspection methods and a rigid infrastructure. A CCW report from 2017 [6] detailed that England and Wales lost a cumulative 3.1 billion litres of water each day through pipeline leaks, up 1.2% from previous years. The scale of the problem derives from current fault detection technologies and methods, causing water companies to adopt a reactive approach to much larger leaks: repairing faults as they develop, as opposed to proactively seeking faults and fixing them before they become problems. They are often unaware of the numerous small leaks certain pipes have, and as such can only respond when large quantities of water are lost. Even then, the process of fixing a leaking pipe is long and painful, usually reacting to customers who have lost water access and shutting down large portions of networks to isolate the source of the problem through digging and pipe replacement. Another difficulty water companies face is that they have insufficient information regarding current networks. Newer networks are often built upon older ones with the current state of neither truly known. As such, without carrying out a complete manual inspection of every network in the UK, it is near-impossible for companies to take a proactive approach to reduce the number of leaks in networks.

Fig. 1 shows the geometry of a real-life water supply network of 560 pipes and a total length of 28.7 kilometres. Much of this network is under residential housing, eliminating invasive inspection processes, whilst the sheer size makes remote-controlled inspection nigh-on impossible.

Fig. 1 A typical complex *looping* network

One solution to inspect this network is with a swarm of autonomous robots. The first purpose of the swarm is mapping – a group of robots will enter the pipeline with the goal of mapping the surrounding network accurately. The second purpose is inspection – the swarm of robots will use mapping information to continuously inspect this network and to relay detailed pipe diagnostics back to the surface. Ultimately, certain robots can be fitted with basic repair capabilities to remove blockages as they begin to develop and to fix smaller leaks.

This paper is a preliminary investigation into the feasibility of the second purpose – the inspection process. Autonomous inspection robots do not require human direction or destructive testing and can be equipped with acoustic and other types of sensors to detect abnormalities within the network. They are able to traverse pipes as small as 10-20 centimetres in diameter and can communicate inspection data back to a hub through acoustic waves. In order to prove practical viability of this pipe inspection technology, real life

Richard Molyneux, Christopher Parrott, and Kirill Horoshenkov are with the University of Sheffield, Mechanical Engineering, United Kingdom (e-mail: rmolyneux2@sheffield.ac.uk).

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

problems such as communication ranges and dynamic water flows need to be considered. This paper aims to demonstrate the adaptability and efficiency of various swarm behaviour algorithms the robots can implement to improve performance. Under strict assumptions, it is proven that even a limited number of robots in a realistic pipe network improve the inspection process by an order of magnitude over any current methods which require human intervention.

II. Literature Review

Compared to other swarm behaviours and solutions, underground pipeline networks provide a unique dilemma with regards to their surrounding environment. Without additional infrastructure adaptations, such as relay points, dense earth severely limits the communication range of the robots and eliminates typical communication technologies such as Wi-Fi or Bluetooth. Acoustic waves have an extended range that can travel through water and pipe material so are able to provide a communication channel, which current robot swarms often rely upon. Dynamic water flows throughout the network provide additional complications – if a pipeline's flow is too fast, it may not be possible for a robot to traverse. Combined with the topological stipulations inherent in most networks such as bottlenecks or outlying fringe pipes, rudimentary algorithms often conclude with robots trapped or clumped together. To counteract these complications, it is necessary to adapt existing algorithms developed for above-ground transport route planning and related applications.

The CPP is a heavily studied topic within graph theory, and hence has a huge back catalogue of relevant literature [5]. The problem originally revolved around a single postman servicing a network of streets delivering post whilst trying to minimise the total time it took to complete a route, always returning to the *depot* node. This problem is highly relevant to the problem of pipe inspection with an autonomous robot swarm. It is possible to rewrite the dilemma as a graph theory problem.

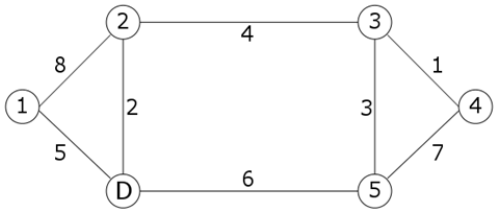

Fig. 2 A simple network set up with a depot node, D, and vertices and *weighted* edges

- A network, or Graph (G), is formed of a group of vertices ($v_i \in V$) and edges ($e_i \in E$).
- An edge e_i can connect either two vertices to one another, $\{v_i, v_j\}$, or a vertex to itself $\{v_i, v_i\}$.
- Edges can have a value associated to them, known as a cost value ($c_i \in C$).
- A 'route' is defined as a walk, W, a finite number of vertices traversed following the associated edges with the associated cost function. So $W = (v_1, v_2, \dots, v_i)$ with the cost $C = (c_1 + c_2 + \cdots + c_{i-1})$.

Therefore, the CPP becomes a simple minimisation problem:

$$min \sum_{i=1}^{k} c_i$$

Fig. 2 shows a network set up in the same format. It is possible to follow multiple routes to complete the network, for which the costs are clearly different:

$$W_1 = (D, 1, 2, D, 2, 3, 5, 3, 4, 5, D)$$
$$C_1 = (5 + 8 + 2 + 2 + 4 + 3 + 3 + 1 + 7 + 6) = 41$$

$$W_2 = (D, 5, 4, 3, 2, 1, D, 5, 3, 2, D)$$
$$C_2 = (6 + 7 + 1 + 4 + 8 + 5 + 6 + 3 + 4 + 2) = 46$$

For the postman, the first route would be the more ideal route to take.

The k-CPP [12] is an extended version of the CPP, wherein multiple postmen service an area. In this instance, ideal solutions to the problem revolve around the postmen planning their paths and working *together*. For example, the most time-consuming method would have $k - 1$ postmen simply remain at the depot node, whilst the remaining postman serviced an area on their own [7]. On the other hand, an optimal algorithm would see the workload distributed relatively evenly with the overall route time decreasing linearly as k, the number of working postmen increases.

TABLE I
Three Separate Combinations of Paths that Three Postmen Could Take to Cover the Network in Fig. 2

Postman 1 $\{W\|C\}$	Postman 2 $\{W\|C\}$	Postman 3 $\{W\|C\}$	Max. C
$\{D,1,2,D\|15\}$	$\{D,2,3,5,D\|15\}$	$\{D,5,3,4,5,D\|23\}$	23
$\{D,1,2,D\|15\}$	$\{D,5,3,5,D\|15\}$	$\{D,2,3,4,5,D\|22\}$	22
$\{D,2,D\|4\}$	$\{D,5,3,5,D\|18\}$	$\{D,1,2,3,4,5,D\|31\}$	31

Evidently the second group of paths in Table I would provide the *least maximum* time, whilst also providing the *least cumulative time*, with different k-CPP solutions targeting a respective minimisation. For example, the first and third solutions have the same cumulative time ($15 + 15 + 23 = 4 + 18 + 31$) however the first solution has a much more even spread of the workload, so would typically be more ideal.

The k-CPP shares similarities with the implementation of swarm robots in pipe networks as in each scenario, multiple agents aim to service the same area in as little time as possible. A noteworthy difference however is that whereas the k-CPP agents must return to the depot node, it is likely that autonomous robots will benefit from continuously inspecting instead. Unfortunately, although the CPP is solvable, the k-CPP is an NP-Hard problem and we are forced to rely on heuristic, near-optimal solutions for our much vaster networks [1].

In order to survey surrounding areas, disciplines such as integer programming or Satellite Navigation (SatNav) employ a 'Branch and Bound' algorithm. Branch and Bound

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

algorithms provide a *complete* method for exploring possible paths from an initial point. For example, SatNavs are able to use their GPS position to determine an initial point, before deploying a Branch and Bound algorithm that will traverse the surrounding area and find road lengths, speed limits and traffic information. This allows the system to enumerate each road with a time value, representing how long it will take to reach each junction, and hence advice the user on the quickest path to a destination by utilising shortest path algorithms such a Dijkstra's or the A* algorithm.

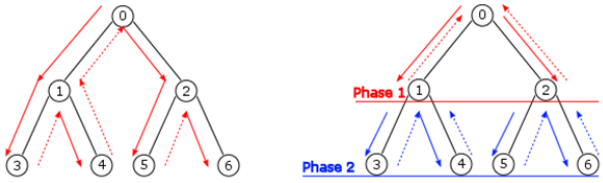

Fig. 3 A Depth First and Breadth First Search navigating a simple environment

There are two main types of Branch and Bound algorithm, Breadth First and Depth First searches (see Fig. 3). A Breadth First Search begins by traversing each vertex that connects to the depot node. It then chooses one of the connecting nodes and traverses its connecting vertices, before returning to another of the first nodes and traversing another set of connecting vertices. Only once each node in a 'level' has been explored will it follow the next nested level. This results in a slow but evenly spread propagation path from the depot node. On the contrary, a Depth First search will follow one path to completion before returning to the last vertex it passed and following that new path to completion. This allows deep exploration in graphs of large scales quickly but does not necessarily explore the depot nodes surrounding areas until the algorithm's conclusion.

To utilise Branch and Bound algorithms in the context of pipeline inspection it is necessary to have a scale by which each pipeline can be given a non-arbitrary number, much like the k-CPP problem. By assigning a value, for example, based on the last time a pipe was inspected, the age of a pipe, pipe length and flow velocity in the pipe, it can be given a *stigmergic* value [4]. To put simply, stigmergy is a virtual representation of pheromones used by foraging ants. Ants leave various pheromones on their trails that convey specific information to other ants that may pass the route, for example whether a food trail is bountiful or barren. Although autonomous robots are not able to leave physical pheromones (yet), it is possible to mimic this behaviour by enabling the robots to exchange information about inspected pipes when they are in close proximity to one another using acoustic communication.

Finally, it is necessary to mention VRP and their adaptations [8]. VRPs are expanded versions of the k-Travelling Salesman Problem (TSP), the counterpart of the k-CPP. Whereas the k-CPP is an *arc*-routing problem, the k-TSP is a *node*-routing problem. Fortunately, by finding a set of minimum-cost vehicle cycles such that the demand serviced by each cycle does not exceed the capacity allows us to treat VRPs as arc-routing problems and hence access the backlog of node-routing literature [9]. For example, the Split Delivery VRP [2] is an adaptation of the problem wherein certain customers have specific requirements and there are specific agents that can fulfil those requirements. In this instance, a known faulty pipe may benefit from additional inspection methods, in which case only robots with the relevant inspection equipment fitted would service that pipe. The VRP with Time Windows [3] provides the additional stipulation that agents can only service specific arcs between 'soft' and 'hard' time targets. Though not implemented here, this remains a viable option to deal with flow changes – if a flow is too strong for an agent to traverse the majority of the time, the agent should only consider it within a specific timeframe.

III. Problem Formulation

To determine the feasibility of autonomous robotic inspection, it is necessary to provide a competitive inspection process for underground pipes. A simple criterion of 'success' is the reduced Time between Inspections (TBI) for as many individual pipes in the network as possible. We expect that pipes on the outer ends of a network are to typically be inspected less, as pipes in the centre are traversed more as the robot agents move to other pipes. Given the topological differences of each network, a good algorithm will be unbiased and adaptive, seeking complete coverage regardless of the networks structure. The TBI values can be normalised with respect to the robot movement speeds and the lengths of the pipe network they cover:

$$T_n = \frac{T_{bi} \times S}{L}$$

where T_n represents the normalised TBI value, T_{bi} the simulation TBI value, S the robot movement speed and L the pipe network length.

The ideal behaviour an algorithm should implement is to have robots work *together* in such a way that pipes that have not been inspected recently are prioritised over those that have. A low mean value of T_n implies an efficient overall inspection process, whereas a low standard deviation in this time indicates that the algorithm provided a more complete coverage.

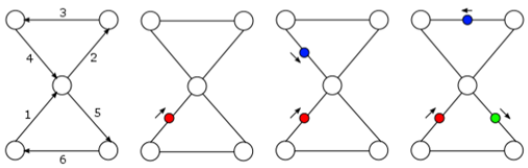

Fig. 4 A simple network's optimal route with an increasing number of Out-of-Sync robots

Unfortunately, although companies record their inspections and maintenance work, no real autonomous inspections or data have been recorded due to its relatively new approach. Although this paper will prove that an autonomous inspection

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

process can increase the frequency of inspections by an order of magnitude in comparison to current methods, a more significant output is determining a baseline standard by which future algorithms can be compared.

Without water flow, it is possible to find an optimal route for a single robot to take to traverse a whole network. By introducing more robots working out-of-sync the frequency at which pipes are inspected increases linearly. This seems like a sensible baseline to which other algorithms can be compared and was trialed early on in the simulation stage. However, this Out-of-Sync Cyclic (O-o-SC) behaviour struggles to adapt when fast flow rates are introduced. This is due to the fact that fast flow rates affect the movement of the robots which, when combined with topological obstacles such as bottlenecks, can cause collisions or loss of control. In addition, this behaviour lacks any kind of swarm cohesion, working together only indirectly to increase the behaviours aptitude. This serves as a good example of a simple behaviour unable to adapt to the network's environment. Within pipeline networks, water flow is the true test of autonomy, and an algorithm that can adapt well to unexpected flow changes will ultimately provide a more comprehensive approach to robot swam control.

Currently, the condition, topology and access points of many underground network arcs are unknown. Similarly, the hardware required for viable autonomous robots is not yet available. This makes a physical demonstration of the algorithms near-impossible and as such this paper details a simulation of virtual robots to illustrate the efficiency of the adopted algorithms and feasibility of the concept that pipes can be inspected continuously with a robot swarm.

IV. Simulation

The simulation has been designed to calculate data, but also to provide a unique insight into the robot behaviour through visualisation. The programme itself was written in the language of C++. Because the simulations parameters, network and robot movement are coded separately to the behaviours, it is impossible to achieve false results as the robots simply follow predetermined instructions within their own constraints.

Fig. 5 A snapshot of seven robots traversing the simulation of the looping network

The simulation takes in real pipeline data translated through the EPAnet software. This allows a two-dimensional render of real-life pipeline networks to be constructed, along with the accompanying water flow data. Robots are then implemented with movement speeds depending on the flow and the ability to communicate via short range acoustic signals.

Simple assumptions are made regarding the robot's capabilities. The robots are inspecting pipes continuously as they move at their reflective movement speed which considers the current water flow in the pipe the robot is currently traversing. The robot's communication range is based on linear fragmentation, and so is not radial. This means that a robot is able to transmit over a set range down a path, regardless of the networks twists and turns, until this range has been exhausted. The robot's computational ability is assumed sufficient enough to compute their future paths. Finally, flow changes hourly, and is not continual, as in accordance with the data provided.

Fig. 5 shows an example of the simulation running. The pipelines (vertical scale) are colour coordinated with respect to the time since they were last inspected, representing their stigmergy value. A light yellow indicates that a pipe was recently inspected whereas the gradual gradient shift to dark red or purple indicates pipes that have not been inspected over relatively longer times.

Fig. 6 shows a *history map*, depicting the colour history for each pipe. Each horizontal line represents a single pipe in the network. The changes in colour correlate to the Time Since Last Inspection seen on the simulation map and change over time. This provides valuable insights into the behaviour's efficiency – an even tone throughout shows even coverage, whereas colour disparity shows that certain pipes are inspected more frequently than others. Similarly, a very yellow map indicates a low average TBI whereas a darker purple map indicates the algorithm is not performing to a satisfactory level.

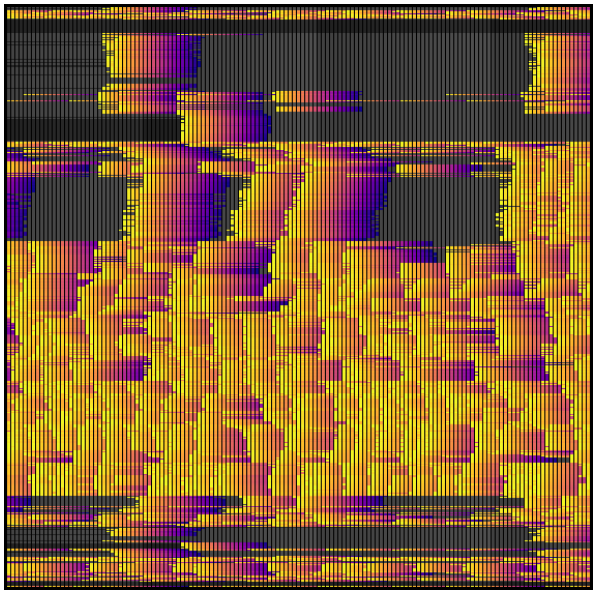

Fig. 6 A render of the history map for the looping network

The simulation is also able to output each robot agent's perceived memory. As detailed in Section V, multiple

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

algorithms depend upon communication, though often not at the same time, leading to disparity between agent's memories or 'individual maps' as illustrated in Fig. 7. If an algorithm does not force frequent communication, there is often disparity between the multiple individual maps.

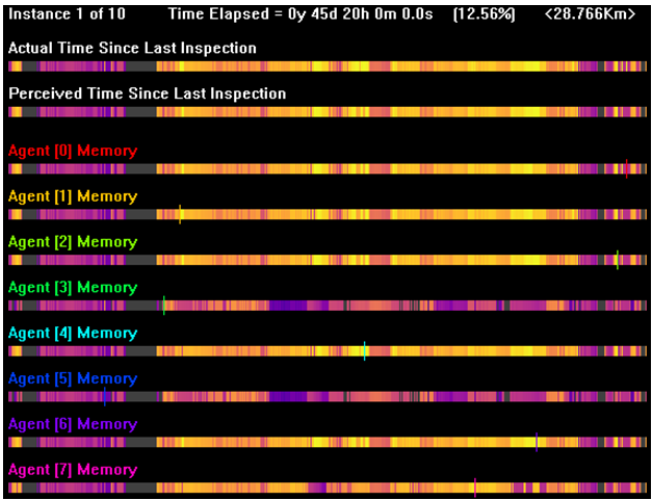

Fig. 7 Individual memories or 'maps' for each robot representing the Time Since Last Inspection

The end of the simulation outputs each time a pipe was inspected along with the robot that inspected it. The data are outputted straight into a Microsoft Excel document for subsequent MATLAB analysis of the mean and standard deviation of T_n.

V. Algorithms

There are three behaviours this paper details, with a range of complexity and efficiency. The first is the simplest, an indirect coordination method similar to that proposed with ants earlier. The second combines a k-CPP heuristic with District Metred Area analysis [11]. The third is a mobile ad-hoc network of robots that traverse the network like a sweeping net [5]. Each of these approaches to the communication and flow issues in different ways and implements varying degrees of autonomy.

The indirect coordination method is surprisingly powerful for such a simple method. The robots traverse a pipe, each with their own stigmergy based memory of the network and the last time each individual pipe was inspected. As a robot reaches a junction, it determines which pipes have been inspected most recently, choosing the *least* recent one. Only once a robot comes within communication range of another robot will it exchange information, updating their maps with the others inspection route. The process is repeated each time another robot is encountered so that there is a continuous process of information exchange between robots and corrections to their respective future routes. By following this rudimentary behaviour, unbiased pipe coverage is achieved as the robot cares little about the topology of the network and is solely dependent on the information regarding the pipeline's value. Unfortunately, clusters of robots can form, following similar paths as their information exchange using this process is out of sync.

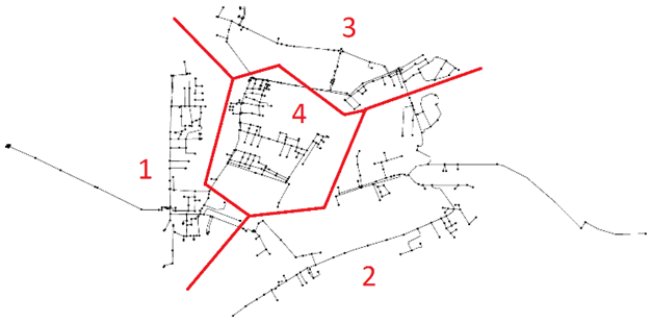

Fig. 8 The looping network split into four DMA regions using cluster analysis

The k-CPP takes advantage of often numerous hydrant points within a network. For example, the network in Fig. 1 has over 20% of its junctions as hydrant points. By splitting a network into regions using a simple cluster analysis [10], it is possible to reduce the pipeline graph into a series of distinct smaller sub-graphs, each with their own depot node (see Fig. 8). Once a robot has been allocated a sub-graph area, it will implement a k-CPP heuristic traversing a much smaller area, reducing computational complexity whilst completely bypassing communication issues by exchanging all information above ground. It is assumed that hydrants provide logical depot/entry points for robots given their surface access, though additional changes to infrastructure could drastically improve the efficiency of this algorithm.

A solution is proposed using the ad-hoc sweeping network of robots. An ad-hoc network is a self-sufficient network that does not require full knowledge of an infrastructure. The core concept of the 'sweeping net' is to plan paths for each robot such that they are within range of another robot at the predetermined time of the next communication exchange, at which point the path for each is planned again. The path planning process uses a rudimentary Depth First Branch and Bound algorithm to find the path combinations that inspect the pipes most 'in need' of inspection. The robots work together to ensure that they are not traversing similar trails around one another, and the whole group tends towards a cyclic net sweeping around the network, always within range of the cluster at the time of the communication exchange. When a robot passes a hydrant, it communicates the most up-to-date map of the network that the robots share. This implementation provides a rudimentary solution to communication problems, whilst allowing complete autonomy and a generally even coverage, in time with the cyclic sweep. The efficiency of the algorithm is highly dependent on the number of robots, increasing exponentially as the number of robots does. In short, a bigger net covers more surface area.

VI. Results

The simulation was run for a 28-day inspection scenario. The results were the T_n mean and standard deviation for each

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

algorithm in two networks of different topologies and sizes, both with and without flow. The first network is from Fig. 1, with the second a much smaller network, nearly a tenth of the size, spanning 2.8 kilometres. Most notably however, the network has a lot fewer topological complexities such as loops and has a much more even spread of junctions (see Fig. 9).

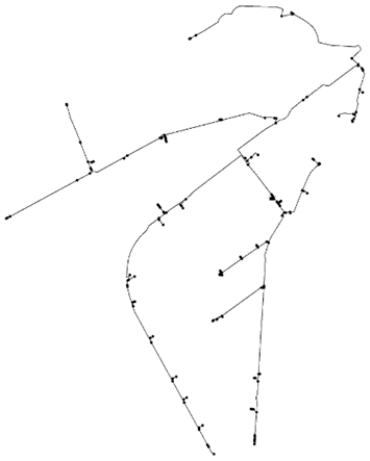

Fig. 9 A simpler, smaller *branching* water pipeline network

The efficiency of these algorithms as a function of the number of robots in the swarm mwas studied. This allowed us to see which algorithms benefit more from the increasing number of agents, or if the work of one algorithm with so many robots can be done the same by another algorithm with less. Each iteration of the simulation (for example, indirect stigmergy behaviour, without flow, on the second network, with 17 robots) was repeated 20 times, with randomly allocated starting points (at a hydrant) to eliminate human bias. The movement speed was set at a realistic 10 centimetres per second whilst communication ranges of 0%, 0.1%, 1%, 10% and 100% of the network size were tested to determine a necessary range for certain algorithms to be effective.

Preliminary tests found that without flow, the k-CPP was a competitive algorithm if the depot points were spread evenly. However, the introduction of flow significantly reduced its efficiency. Typically, the agents leave a depot node at the same time, having exchanged their information and planned their routes prior to leaving. In underground networks, without considering a 'future flow' it is highly likely that certain robots can get stuck. Unlike other behaviours, the robots do not recompute a different path under these conditions, and the agents already at the depot node will wait to leave again until all the agents are back. As such, when extreme delays occur (a path in the looping network has such a frequent fast flow that it often traps a robot for five or more days), it results in k robots out of action, as opposed to the single one in other algorithms. Instead, an adaptation focused on Multiple CPP solutions is proposed for future work whilst the remaining results detail how the stigmergy and ad-hoc behaviours adapt to flow and various communication ranges.

To implement a Multiple CPP problem it is necessary to reduce the DMA cluster sizes. In this scenario, the number of clusters is equal to the number of robots, each with their own individual cluster. This way, an exact CPP solution can be implemented with the usual advantage of not requiring communication. Given a likely abundance of hydrants in most systems, it is natural to assign a robot to an area encompassing multiple hydrants; however there are scenarios in which hydrants or communication relay points are limited. Only if the number of robots outweighs the number of these depot points should certain regions share robots and implement a k-CPP heuristic instead. In these situations, it is possible to normalise the TBI between regions by increasing the number of robots in a region that is underperforming.

Figs. 10-13 illustrate the changing performances of the stigmergy and ad-hoc behaviours as a function of the number of robots (mean – left, standard deviation – right).

VII. Analysis

The most basic observation is that, as expected, the introduction of additional robots reduces the mean and standard deviation for both networks. This is shown by the generally linear negative gradient and is a clear indication of an improvement in performance as more robots are introduced. The different gradients represent a measure of benefit a certain algorithm gains with additional robots, and gives an indication of a reliance on a 'sufficient' number of robots for each algorithm.

A second general observation is that, as expected, flow severely hinders the rate at which a swarm is able to inspect the network. For a smaller number of robots, the mean and standard deviations differ only slightly from a network without flow. However, as additional robots are introduced, we see that both the mean and standard deviations remain at higher T_n values than their without-flow counterparts. For the higher number of robots, this could be simply due to the fact that there is often overlap in the paths robots will take. When combined with fast flows, this can result in clumping – for a period of time multiple robots perform as if they were one robot. However, the noise and slight curves in the looping network with flow, and the vastly different gradients in the branching network with flow, indicate that flow is having a very real effect on the coordination and path-planning processes.

The topography of the networks seems to have a much greater effect on certain algorithms than originally anticipated. As T_n and the communication ranges are normalised with respect to the size of the network, any differences between the two networks are a sole result of the topology of the networks. Though each follows the same general trends with regards to additional robots and flow changes, the better performing algorithms differ completely. Whereas the looping network is inspected at a better rate by stigmergy behaviours of various ranges, with the sole exception of ad-hoc with *complete* communication range (red line under the black), the branching network finds that the ad-hoc behaviour tends to outperform stigmergy. The looping network has a much denser concentration of clusters and cycles whereas the branching

network is much more one dimensional. This suggests that in terms of topology, the ad-hoc behaviour is better when the network is simpler. On the contrary, the stigmergy behaviour seems more adaptable to topological stipulations.

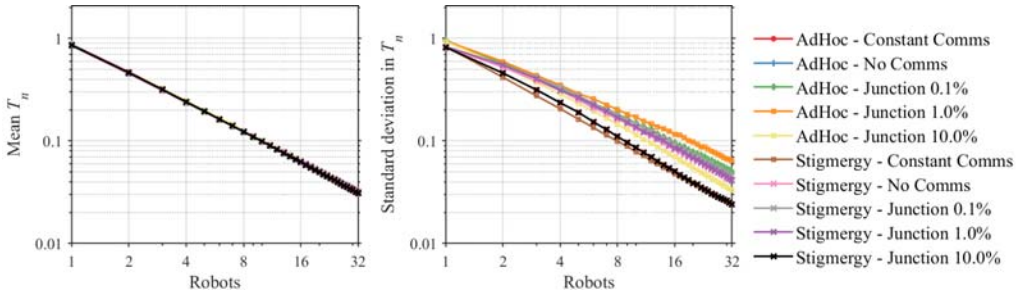

Fig. 10 Looping network without flow

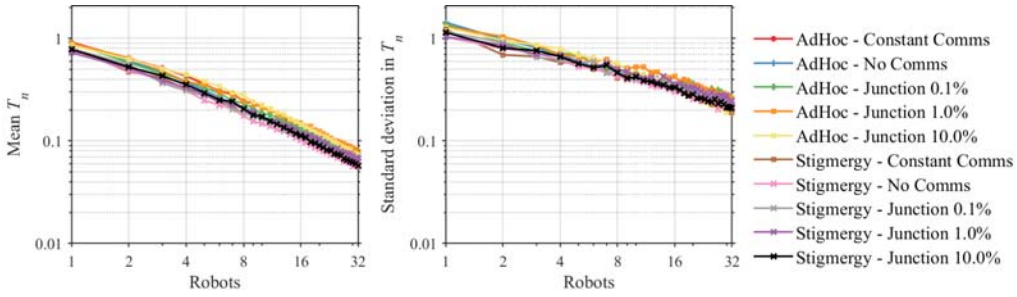

Fig. 11 Looping network with flow

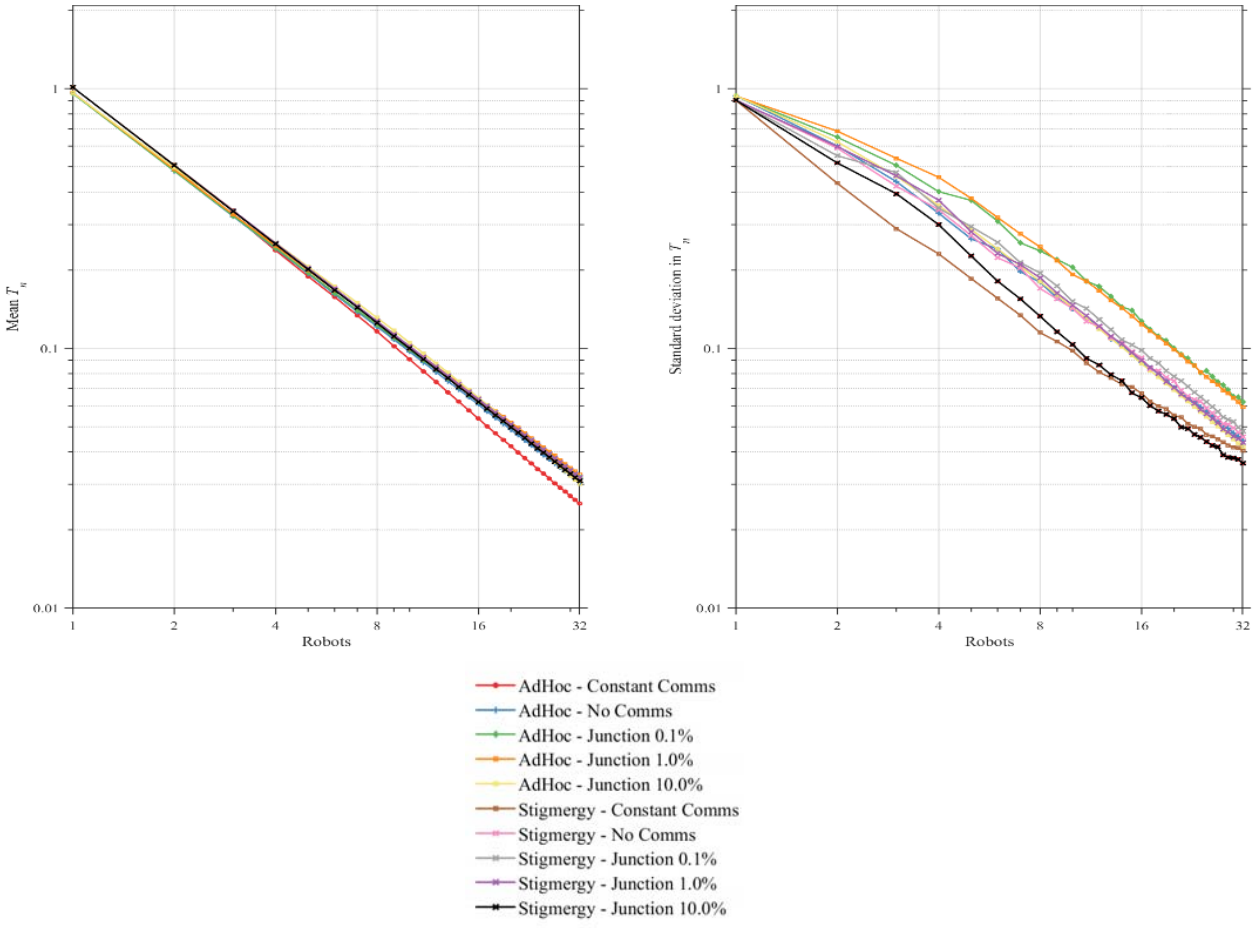

Fig. 12 Branching network without flow

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

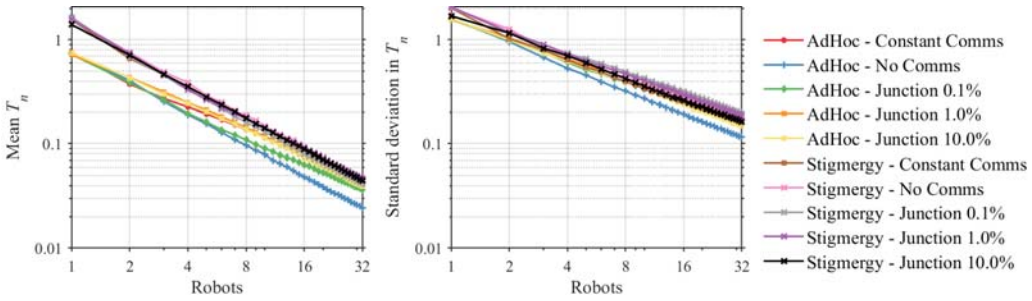

Fig. 13 Branching network with flow

In terms of the algorithms themselves, the ad-hoc behaviour appears to offer a lower mean T_n value whereas the indirect coordination method tends to have a low standard deviation. This implies that the stigmergy behaviour tends to reach *all* paths at a more infrequent rate, whereas the ad-hoc behaviour tends to inspect a smaller range much more frequently. The history map (see example in Fig. 6) of the networks showed the ad-hoc behaviour inspected central pipes much more frequently as it planned paths through them to reach other in-need pipes but as a result, often neglected outliers or satellite pipes. On the other hand, the stigmergy behaviour is unbiased to the direction it tends towards and, as previously mentioned, is more capable of dealing with topological obstacles such as outliers. Therefore, we see that the behaviour inspects fringe pipelines much more often at the cost of a lower mean.

With respect to communication ranges, the stigmergy behaviour follows the expected trend – larger communication ranges yield a lower T_n value. This is seen in both networks, both with and without flow and stems from the fact that an increased range increases the amount of information an agent has regarding its immediate surroundings. As the stigmergy behaviour is solely dependent on this information, the expected order is preserved. A similar pattern is followed for the ad-hoc behaviour in the looping network; however, a significant difference is observed in the branching network. It is evident that with flow, a zero communication (blue), or tiny communication (green) range is beneficial. Interestingly, without flow the network sees that continual constant communication (red) provides the most complete coverage whilst zero communication is mediocre. The switch between these two once flow is introduced indicates that in simple networks there is no necessity to work together *directly*. Despite this, the mean values for the ad-hoc behaviour in this network are all significantly lower which implies that some measure of planning is still beneficial. We are therefore led to believe that the benefits of the zero or small communication cases come from the specific combination of the topography of the branching network and the introduction of flow – the no communication case has a very linear mean and standard deviation plot possibly indicating that it has unintentionally mimicked the Out-of-Sync Cyclic behaviour mentioned previously. With a lack of communication, the agents are unable to function as a net, so the behaviour instead proceeds as individual robots planning future paths individually. As we examine the topology in the branching network it is evident that this could in turn become a cyclic method, iteratively inspecting the four main branches from the central area. In this situation, a single robot's behaviour would become cyclic, and a small communication range ensures the very little is done to affect this balance. Due to the behaviour's ability to re-plan a path if it cannot overcome the flow, it is conceivable that the agents may have coincidentally performed an effective O-o-SC behaviour that is capable of adapting to flow.

Another explanation can be found by examining the crossing points in the ad-hoc behaviour. In the looping network, the gradients of both algorithms appear to be similar, and though the performance is different, it is clear that both gain a similar level of benefit from increasing the number of robots. In the branching network, we see that the performance of the 0.1% communication range suffers as more robots are introduced whereas the zero-communication condition follows the ideal linear progression. This may be due to a fault in the algorithm – currently robots will aim to avoid following paths that another robot is planning to traverse. It is possible that as the number of robots increases, the map becomes too saturated and the robots become trapped by other robots. We see a divergent point between the 0.1% and zero-communication cases after the sixth robot, the point at which it would appear additional robots begin to over-saturate the network. In this instance it is possible to hypothesise that as the number of robots increases further in the looping network, at some point there will be a crossing point wherein the lower levels of communication surpass those of a greater range.

VIII. Conclusion

The aim of this paper was to demonstrate that a swarm of robots can inspect a realistic underground pipeline network autonomously. A preliminary investigation into the aptitude of multiple algorithms has highlighted the difficulties faced in this particular environment, and the pitfalls that future adaptations must overcome. Furthermore, the paper has demonstrated an improved efficiency for algorithms with an extended communication range, especially for lower numbers of robots, and an overall enhanced performance from an increased number of robots. An unexpected outcome of the analysis has been the noted dependency on the topology of a network and further research must be done to determine the qualities certain algorithms will thrive in over others. Despite a loss in performance, the simulation platform has demonstrated that multiple behaviours are able to overcome

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712

fast water flows and still provide comprehensive coverage of entire networks. To summarise, the presented results demonstrate the proposed that implementation of autonomous swarms of inspection robots in underground pipeline networks is a viable and competitive method, able to improve the inspection process in an innovative and adaptive manner.

REFERENCES

[1] Ahr, D., Reinelt, G., (2006). A tabu search algorithm for the min-max k-Chinese Postman Problem. *Computers and Operations Research*. 33, 3403 – 3422.

[2] Aleman, R. E., Zhang, X., Hill, R. R., (2008). An adaptive memory algorithm for the split delivery vehicle routing problem. *Springer Science and Business Media – Heuristics*. 16, 441 – 473.

[3] Desrochers, M., Desrosiers, J., Solomon, M., (1992). A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows. *Operations Research*. 40, 199 – 415.

[4] Dorigo, M., Bonabeau, E., Theraulaz, G., (2000). Ant algorithms and stigmergy. *Future Generation Computer Systems* (online). Volume 16, 851-871. (Last viewed 29/06/2019). Available from: DOI: 10.1016/S0167-739X(00)00042-X. ISSN: 0167-739X.

[5] Johnson, D. B., Maltz, D. A., (2012). *Dynamic Source Routing in Ad Hoc Wireless Networks*. The Kluwer International Series in Engineering and Computer Science book series (SECS, volume 353).

[6] Loughran, J., (2017). *Water leakage from UK pipes rises to over three billion litres a day* (online). Engineering and Technology (E&T) (Last viewed 29/06/2019). Available from: https://eandt.theiet.org/content/articles/2017/12/water-leakage-from-uk-pipes-rises-to-over-three-billion-litres-a-day/

[7] Osterhause, A., Mariak, F., (2005). On variants of the k-Chinese Postman Problem. *Operations Research and Wirtschaftsinformatik*, 30.

[8] Paraskevopoulos, D. C., et al., (2017). Resource constrained routing and scheduling: Review and research prospects. *European Journal of Operations Research*. 263, 737 – 754.

[9] Pearn, W., Assad, A., Golden, B. L., (1987). *Transforming Arc Routing into Node Routing Problems*. Great Britain: Pergamon Journals. 14, 285-288.

[10] Perelman, L., Ostfeld, A., (2012). Water-Distribution Systems Simplifications through Clustering. *American Society of Civil Engineers* (online). Volume 138 (3). (Last viewed 29/06/19). Available from: DOI; 10.1061/(ACSE)WR.1943-5452.0000173.

[11] Scarpa, F., Lobba, A., Becciu, G., (2016). Elementary DMA Design of Looped Water Distribution Networks with Multiple Sources. *American Society of Civil Engineers* (online). Volume 142 (6). (Last viewed 29/06/19). Available from: DOI: 10.1061/(ACSE)WR.1943-5452.0000639.

[12] Thimbleby, H., (2003). The directed Chinese Postman Problem. *Software: Practice and Experience*. 33, 1081 – 1096.

Open Science Index, Mechanical and Materials Engineering Vol:13, No:9, 2019 waset.org/Publication/10010712