IR-XY-PV: Infrared Multihop Communication for XY-Coordinated PV Modules

Hideya Ochiai
The University of Tokyo
ochiai@elab.ic.i.u-tokyo.ac.jp

Zhiqing Zhang
Tsinghua University
zhangzq14@mails.tsinghua.edu.cn

Hiroto Kitamura
The University of Tokyo
zun@hongo.wide.ad.jp

Hiroshi Esaki
The University of Tokyo
hiroshi@wide.ad.jp

ABSTRACT

Photovoltaic (PV) power stations are rapidly increasing as an alternative energy resources for oil, natural gas, coal, and nuclear. If each PV module has intelligence and the ability to report its working status (e.g., voltage and temperature), we can manage the system status of such PV power stations with IoT systems. PV modules are usually installed on an XY-grid, indicating that we can apply an XY multihop routing for gathering such sensor readings with tiny wireless nodes. We propose the architecture and routing schemes of an infrared multihop communication for XY-coordinated PV modules (IR-XY-PV). This includes neighbor discovery and disruption tolerant packet forwarding, i.e., single-copy forwarding and multi-copy forwarding schemes for packet propagation in the network. We have developed 20 node scale IR-XY-PV network and confirmed that IR-XY-PV can provide practically enough performance regarding delivery success rate, delivery latency and memory usage.

CCS CONCEPTS

• **Networks** → **Layering**; **Network protocol design**; *Sensor networks*;

KEYWORDS

Multihop Communication, IoT, PV, DTN

ACM Reference Format:
Hideya Ochiai, Zhiqing Zhang, Hiroto Kitamura, and Hiroshi Esaki. 2017. IR-XY-PV: Infrared Multihop Communication for XY-Coordinated PV Modules. In *AINTEC '17: AINTEC '17: Asian Internet Engineering Conference , November 20–22, 2017, Bangkok, Thailand*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3154970.3154971

1 INTRODUCTION

Management of photovoltaic (PV) power stations is one of the important applications of the Internet of Things (IoT) these days [3, 6, 13]. A PV power station has tens of, hundreds of, or even thousands of PV modules depending on its scale. Some of them sometimes become malfunctioned causing fatal performance degradation, which can be detected at the macro-level – but it is not easy to identify the wrong module(s) without checking each PV module one-by-one. If each PV has intelligence and the ability to report its status (e.g., voltage and temperature), todays IoT systems can tell the identified wrong PV panels to their remote operators [7, 9–11].

PV modules are usually installed in an XY-coordinated manner, indicating that we can deploy wireless tiny sensor nodes on each PV module and can run an XY distance based routing for sensor data gathering. We can probably design a sensor node using infrared LED as a transmitter and photodiode as a receiver assuming to attach itself on a PV module in the manufacturing process.

In this paper, we propose the architecture and routing schemes of an infrared multihop communication for XY-coordinated PV modules (IR-XY-PV). Each PV module has an addess on an XY-grid, e.g., (s_x, s_y), and can send packets to a destination address (d_x, d_y). The packet may contain the voltage information which can be used for PV module diagnosis. In this work, we identify the network architecture and study XY routing for a narrow-band infrared communication network. They include neighbor discovery and disruption tolerant forwarding schemes.

"XY routing" performs packet routing by choosing the closer node to the destination as the next hop node. Here, the distance metric is defined on the XY-grid. It is basically stateless. An intermediate node (i.e., router node) does not have a stateful routing table as the Internet protocol (IP) routers do. However, each node still (1) needs to know the existence or availability of neighbors, (2) needs to confirm that their forwarded packet has been accepted by the next hop node in order to perform disruption tolerant forwarding, and (3) needs to manage its packet buffer. In this paper, we design the network architecture with considering these issues.

We also study the packet forwarding schemes – single-copy forwarding (SCF) and multi-copy forwarding (MCF). SCF just makes one copy of a packet in the network during the packet delivery, whereas MCF generates many copies in the network so as to have multiple delivery paths for redundancy and faster delivery.

We have developed 20 IR-transceiver nodes and evaluated (1) delivery success rate, (2) delivery latency, (3) delivery paths, and (4) buffer occupancy for SCF and MCF with different distance cost metrics and network parameters. Clock synchronization, configuration of node address, and packet broadcasting are also important but we do not focus these issues in this paper.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
AINTEC '17, November 20–22, 2017, Bangkok, Thailand
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-5551-3/17/11…$15.00
https://doi.org/10.1145/3154970.3154971

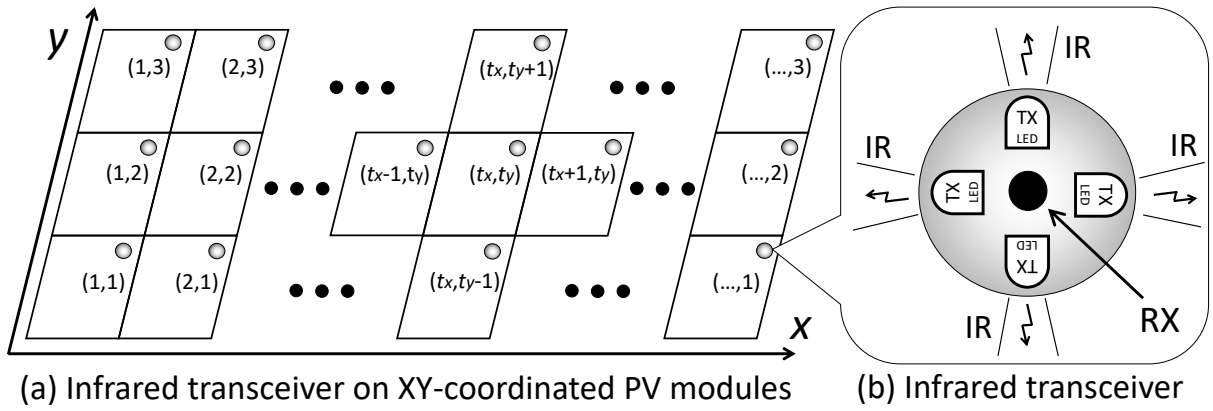


Figure 1: IR-XY-PV deployment plan – IR transceiver addressing for XY-coordinated PV modules

This paper is organized as follows. Section 2 provides the related work. We propose IR-XY-PV architecture and its routing schemes in section 3. Section 4 provides our evaluation work with our network testbed. Section 5 gives discussion, and we conclude this paper in section 6.

2 RELATED WORK

Module level PV monitoring has been proposed with short-range radio-based wireless communication [10, 11] and power-line communication [7, 9, 12]. In this paper, as an alternative method, we are proposing infrared communication for PV module monitoring.

XY routing, we propose in this paper, fits into the family of position-based routing or geographical routing [8]. However, most of the position-based routing were discussed in the context of vehicular ad hoc networks (VANETs) [1], where the nodes move on XY-plane. Hybrid with delay/disruption tolerant network (DTN) [4, 14] were also for VANET.

A number of DTN routing schemes on narrow-band and intermittently connected links have been studied [2]. However, it is not studied on PV module networking, especially with infrared and XY routing.

Optical wireless (OW) communication [5] are well-studied, however, we have not found the work of multihop network with infrared sensor nodes.

The contribution of this paper is to propose the network architecture and routing protocol of an infrared based XY routing for PV module networking.

3 INFRARED XY-ROUTING FOR PVS

This section describes the architecture and routing schemes for infrared multihop communication for XY-coordinated PVs (IR-XY-PV). As Fig.1, we attach an infrared communication node on the surface of each PV module, assigning the address of a XY-grid (i.e., $\mathbb{Z} \times \mathbb{Z}$). The communication node has sensors with which it can measure the status of the PV module (e.g., voltage and temperature). Each node, for sending those sensor readings, periodically generates a packet p, which contains:

- $p.dst$: The destination node of the packet, which we denote by $(d_x, d_y) \in \mathbb{Z}^2$.
- $p.src$: The source node of the packet, which we denote by $(s_x, s_y) \in \mathbb{Z}^2$.
- $p.time$: The time of generating the observation values.
- $p.values$: The observation values (e.g., voltage and temperature of the PV module).
- $p.crc$: The checksum of this packet.

The source node sends p to the routing layer, which delivers p to the destination $p.dst$. Then, the destination node will receive p and use the contents for their application after verifying p with $p.crc$. In IR-XY-PV, we considers p_0 and p_1 are identical if all the parameters of p_0 and p_1 are the same.

In the following discussion, with Fig.2, we focus on the routing scheme of IR-XY-PV. It contains the design of (a) IR transceiver node, (2) neighbor discovery, (3) disruption tolerant XY routing with SCF and MCF.

We introduce the concept of "this" node (which we denote by $\mathbf{t} = (t_x, t_y) \in \mathbb{Z}^2$) for desiging such routing schemes. We discuss, focusing on node $\mathbf{t}$, for discovering neighbors and for forwarding a packet to the next hop.

3.1 IR Transceiver Node

In our design, the transceiver has four IR-LEDs for transmission and one photodiode for receival as Fig.1(b). These LEDs can be controlled separately. For example, only the left LED can be used for (t_x, t_y) to transmit data frame to $(t_x - 1, t_y)$. Though there may be several design choices for IR physical communication protocols, in this paper we consider to use 38kHz carrier based IR signal, as they are widely used as worldwide standard for short-range and easy-to-use communication. It encodes bit 0 and 1 for different duration of signal transmission as Fig.2. The starting mark has longer duration for the receiver to identify the start bit.

3.2 Neighbor Discovery

As Fig.2, node $\mathbf{t}$ has a neighbor map at the routing layer. In IR-XY-PV, they run a neighbor discovery protocol and manages the list of neighbors. This includes the statistics calculation.

Instead of broadcasting a discovery frame and waiting for responses from all the possible neighbors, IR-XY-PV takes master/slave approach – node **t** sends the discovery request frame for potential neighbors one-by-one. This is possible because nodes are deployed on a XY-grid and neighbor's addresses can be calculated.

The following algorithm describes neighbor discovery routine at node **t**:

Function NeighborDiscovery()
  For $i = 1$ to MAXRAD **do**
    NDReq($(t_x - i, t_y)$)
    NDReq($(t_x, t_y - i)$)
    NDReq($(t_x + i, t_y)$)
    NDReq($(t_x, t_y + i)$)
  End For
End Function

Here, MAXRAD is a constant parameter that represents the maximum radius for single hop transmission. In each NDReq(**n**) process, node **t** sends neighbor discovery request to node **n**. Node **n** then returns neighbor discovery response to node **t**. If node **t** does not receive the response within a certain duration (e.g., within 1 second), **t** sends the request again. **t** stops sending the request after receiving the response or after trying three times.

Each node carries out this algorithm periodically, for example, with 300 second interval. Based on the responses of neighbor discovery requests, node **t** generates the neighbor map in the following manner. Neighbor map consists of the list of (1) neighbor node, (2) link cost, and (3) node cost.

Let $L_C(\mathbf{n}, \mathbf{t})$ be a cost of link that represents the link quality from **t** to **n**, and $TryCount(\mathbf{n}, \mathbf{t})$ be the tried count of neighbor discovery request from **t** to **n**. After NDReq(**n**) is finished, node **t** updates $L_C(\mathbf{n}, \mathbf{t})$ with the following rule:

$$L_C(\mathbf{n}, \mathbf{t}) := a \cdot L_C(\mathbf{n}, \mathbf{t}) + (1 - a) \cdot TryCount(\mathbf{n}, \mathbf{t}) \quad (1)$$

Here, a is a smoothing parameter, which is given as a constant value between 0 and 1. If NDReq(**n**) is finished with failure (i.e., the neighbor response is not received in the three trial), we set $TryCount(\mathbf{n}, \mathbf{t})$ to be five as a special case.

The neighbor map also manages the cost of node **n** (which we denote by $N_C(\mathbf{n}, \mathbf{t})$) from the observation by **t**,

$$N_C(\mathbf{n}, \mathbf{t}) := \begin{cases} b & \text{if } \mathrm{F}_{\mathrm{reject}} \text{ received,} \\ c \cdot N_C(\mathbf{n}, \mathbf{t}) & \text{otherwise.} \end{cases} \quad (2)$$

Here, b is a positive constant value which represents the maximum cost of the node. $N_C(\mathbf{n}, \mathbf{t})$ will be set to b when $\mathrm{F}_{\mathrm{reject}}$ is received from **n** at **t** (as for $\mathrm{F}_{\mathrm{reject}}$ – see Section 3.3). c is a smoothing parameter, which is given as a constant value between 0 and 1.

3.3 Disruption Tolerant XY Routing

As Fig.2 shows, node **t** has a packet buffer and a history buffer. The packet buffer stores packets for achieving disruption tolerant store-and-forward scheme. The history buffer manages the footprint of the packets in order to tell to the previous hop node that this node has already had the packet in the near past.

The packet buffer is a table of

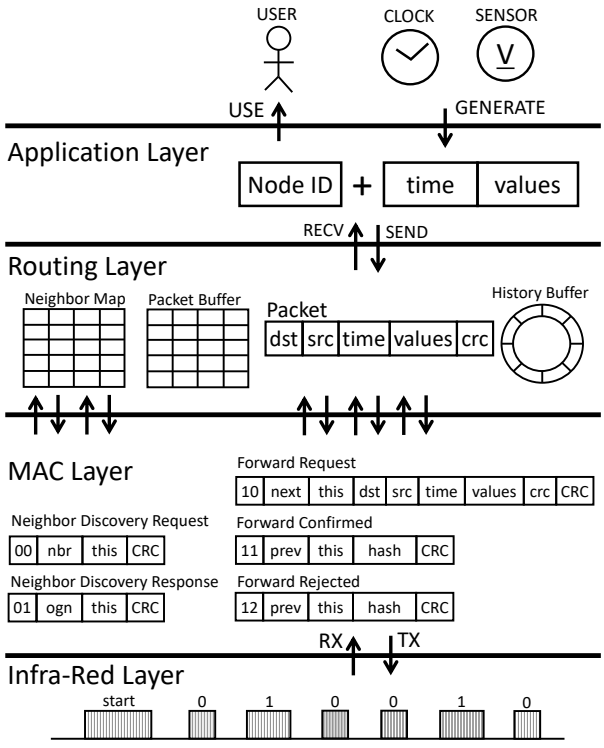


Figure 2: IR-XY-PV protocol stack: (1) application layer generates and uses packets, (2) routing layer delivers packets to the destination node, (3) MAC layer manages the communication with neighbors, and (4) infrared layer modulates and demodulates data bits.

- packet: key of the table,
- lst: last sent time of the packet,
- fn: forwarded nodes of the packet,
- fr: forward retry count remains for the packet,
- at: timestamp of the packet arrival.

Those parameters are bound to each packet, so we describe them for instance of packet p as $p.lst$, $p.fn$, $p.fr$, $p.rt$ in the following discussion. When a new packet p arrives at **t**, the packet buffer will add a new record for p initializing that $p.lst = 0$, $p.fn$ =NULL, $p.fr$ =INFINITE, and $p.at$ =time(). Here, time() gives the current timestamp.

For forwarding a packet between nodes, IR-XY-PV defines the following three data frames as Fig.2: (1) forward request ($\mathrm{F}_{\mathrm{request}}$), (2) forward confirmed ($\mathrm{F}_{\mathrm{confirmed}}$), and (3) forward rejected ($\mathrm{F}_{\mathrm{rejected}}$). Node **t** sends a packet on $\mathrm{F}_{\mathrm{request}}$ to the specified next **n**, and then **n** replies back $\mathrm{F}_{\mathrm{confirmed}}$ or $\mathrm{F}_{\mathrm{rejected}}$ to the previous node, i.e., **t**. $\mathrm{F}_{\mathrm{confirmed}}$ means that the packet has been accepted to the node, whereas $\mathrm{F}_{\mathrm{rejected}}$ means the packet has not been accepted by the node. $\mathrm{F}_{\mathrm{confirmed}}$ and $\mathrm{F}_{\mathrm{rejected}}$ do not contain the packet itself. Instead, they contain the hashed value of the packet, i.e., hash(p).

To forward a packet in disruption-tolerant manner, we define two algorithms for single-copy forwarding (SCF) and multi-copy forwarding (MCF).

3.3.1 Single-Copy Forwarding Algorithm. When node **t** has packet p, **t** chooses a single next hop node for forwarding p. In the following discussion, we denote the next hop node chosen by single-copy strategy by $\mathbf{next}_{\mathbf{SCF}}(\mathbf{t}, p)$. For choosing this, we define a cost function for sending p for a next node **n** at node **t** as follows:

$$cost(\mathbf{n}, \mathbf{t}, p) = D(\mathbf{n}, p.dst) + \alpha L_C(\mathbf{n}, \mathbf{t}) + \beta N_C(\mathbf{n}, \mathbf{t}). \quad (3)$$

Here, α and β are positive constants. $D(\mathbf{n}, p.dst)$ gives the distance on the XY-grid from the next **n** to the destination $p.dst$. We included L_C and N_C in the cost function because the network status cannot be simply determined by the distance function but availability of the links and the nodes are also important for choosing the next hop node. Here, we can consider Manhattan distance D_M or Euclidean distance D_E, which are formally defined as:

$$D_M(\mathbf{n}, \mathbf{d}) = |n_x - d_x| + |n_y - d_y| \quad (4)$$

$$D_E(\mathbf{n}, \mathbf{d}) = \sqrt{(n_x - d_x)^2 + (n_y - d_y)^2} \quad (5)$$

In SCF, node **t** chooses the lowest cost node as the next. More formally,

$$\mathbf{next}_{\mathbf{SCF}}(\mathbf{t}, p) = \mathrm{argmin}_{\mathbf{n} \in \mathbf{nbr(t)}}\{cost(\mathbf{n}, \mathbf{t}, p)\}. \quad (6)$$

When $\mathbf{next}_{\mathbf{SCF}}(\mathbf{t}, p)$ is determined, node **t** sends $\mathrm{F}_{\mathrm{request}}$ with p to $\mathbf{next}_{\mathbf{SCF}}(\mathbf{t}, p)$. If the $\mathbf{next}_{\mathbf{SCF}}(\mathbf{t}, p)$ receives the $\mathrm{F}_{\mathrm{request}}$, it replies back $\mathrm{F}_{\mathrm{confirmed}}$ or $\mathrm{F}_{\mathrm{rejected}}$.

If node **t** receives $\mathrm{F}_{\mathrm{confirmed}}$ for p, it removes p from the packet buffer, calculates hash(p) and pushes it into the history buffer. If the history buffer is full, it overrides the oldest one.

If **t** does not receive $\mathrm{F}_{\mathrm{confirmed}}$ or does receive $\mathrm{F}_{\mathrm{rejected}}$, it will update $p.lst$ as:

$$p.lst := \mathrm{time}() \quad (7)$$

Especially if **t** receives $\mathrm{F}_{\mathrm{rejected}}$, it also changes the neighbor buffer based on Eqn. 2.

Node **t** tries forwarding of p if $p.lst$+RFI < time(). Here, RFI is a re-forward interval, which is defined in disruption tolerant forwarding context.

When node **t** receives packet p with $\mathrm{F}_{\mathrm{request}}$ from another node, it first searches the packet buffer whether the same packet is already managed, if so, it just responds $\mathrm{F}_{\mathrm{confirmed}}$. If p is not contained in the packet buffer, it calculates hash(p) and searches the history buffer. If hash(p) is contained in the history buffer, it also just responds $\mathrm{F}_{\mathrm{confirmd}}$. If hash($p$) is not contained in the history buffer, and the packet buffer has a space for accepting, it responds $\mathrm{F}_{\mathrm{confirmed}}$ and store p into the packet buffer as a new packet. It responds $\mathrm{F}_{\mathrm{rejected}}$ if the forwarding decision is not appropriate, for example, if it has no memory space for the packet.

3.3.2 Multi-Copy Forwarding Algorithm. Making multiple-copies in the network will potentially improve the delivery latency and the probability of reachability. Instead, it will have more chance of encountering dead situation. Dead packets will remain in the network, consuming buffer and transmissions. To avoid this, we allow the nodes to intentionally delete the packets which have gone to such situations.

Using the $cost(\mathbf{n}, \mathbf{t}, p)$ defined in section 3.3.1, in MCF, node **t** chooses the lowest N nodes as the next nodes (where $N > 1$); we denote them by $\mathbf{next}_{\mathbf{MCF}}(\mathbf{t}, p)$. Then, it sends $\mathrm{F}_{\mathrm{request}}$ to all of $\mathbf{next}_{\mathbf{MCF}}(\mathbf{t}, p)$ one-by-one except the nodes listed in $p.fn$. The

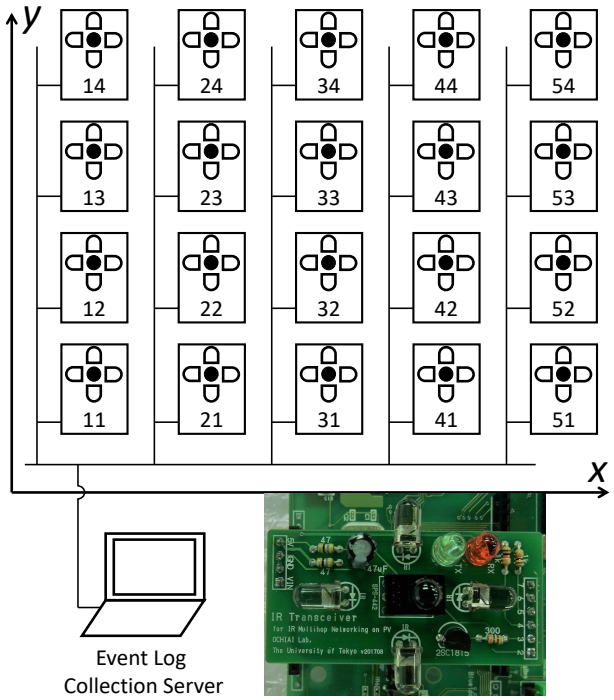


Figure 3: IR-XY-PV experiment with 5×4 deployment configuration. Each node had a management interface through which we collected experiment logs for analysis.

nodes that received $\mathrm{F}_{\mathrm{request}}$ return $\mathrm{F}_{\mathrm{confirmed}}$. Node **t** add the responded nodes to $p.fn$. If the $count(p.fn)$ reaches N, node **t** removes packet p from the packet buffer, pushing hash(p) to the history buffer.

In MCF, we also define intentional packet deletion scheme as follows.

(1) Node **t** removes p if the packet buffer is full and p is the oldest in the buffer when adding a new packet into the buffer. This time, **t** does not return $\mathrm{F}_{\mathrm{rejected}}$.
(2) Node **t** removes p if **t** has received $\mathrm{F}_{\mathrm{rejected}}$ five times from the next nodes.
(3) Node **t** removes p if no more appropriate next candidates available. For example, $degree(\mathbf{t})$ is just one or two.
(4) Node **t** sets $p.fr = 3$ if **t** has received $\mathrm{F}_{\mathrm{confirmed}}$ from at least one of the next nodes. $p.fr$ decrements at every RFI. If it reaches 0, node **t** removes p.

When node **t** removes p from the packet buffer, it adds hash(p) to the history buffer.

4 EVALUATION

We have evaluated IR-XY-PV regarding to (1) packet delivery success rate, (2) delivery latency, (3) delivery path, and (4) buffer occupancy. We have tested with 20 IR transceivers for SCF and MCF, and for different distance metrics.

Table 1: Experiment settings

Type	Parameter	Value
Neighbor Discovery	a	0.9
	b	3
	c	0.9
	MAXRAD	1
	INTERVAL	300
	MAX_RETRY	3
	RETRY_INTERVAL	2-2.8
Packet Forwarding	β	0.33
	CHECK_INTERVAL	2
	RFI	15
	RETRY_INTERVAL	2-2.8
	BUF_SIZE	20
	HASH_BUF_SIZE	16

Table 2: Delivery success rate

Case			Sent	Received
SCF	Manhattan	$\alpha = 0.01$	118	118
		$\alpha = 0.33$	118	118
	Euclidean	$\alpha = 0.01$	118	118
		$\alpha = 0.33$	118	118
MCF	Manhattan	$\alpha = 0.01$	118	117
		$\alpha = 0.33$	118	118
	Euclidean	$\alpha = 0.01$	118	118
		$\alpha = 0.33$	118	118

4.1 Experiment Settings

We developed 20 IR transceivers with Arduino Mega 2560, and deployed them on 5×4 grid as shown in Fig.3. Each node had four LEDs (for left, right, up and down transmissions) and one photodiode for receival. Each node also had an Ethernet port through which we collected log of events for analysis.

We evaluated (1) delivery success rate, (2) latency, (3) paths, and (4) buffer occupancy for Manhattan distance and Euclidean distance, and $\alpha = 0.01$ and $\alpha = 0.33$ for both SCF and MCF schemes. If $\alpha = 0.01$, the distance metric becomes more dominant for choosing the next. If $\alpha = 0.33$, it is more awared of link status for choosing the next. In this experiment, node (5,4) sent packet to (1,1) every 60 second, and we studied the delivery features of the packet. Table 1 shows the other configuration parameters. TX-LED direction control was not applied.

We conducted the experiments in the following manner – (1) we installed software into the 20 nodes, (2) we reset all the nodes, (3) we waited for 15 minutes for the network to be stabilized, (4) 2 hour experiment followed, and (5) we waited addtional 15 minutes for all the packets in the network to be delivered. And, we analyzed for the 2 hour duration.

4.2 Delivery Success Rate

As Table 2 summarizes almost all the packets were delivered in all the cases except one packet was lost in MCF with Manhattan distance and $\alpha = 0.01$. In our observation, this loss was caused by the intensional deletion defined in MCF scheme. All the packet copies were considered to be faced with dead situation and deleted before arriving at the destination.

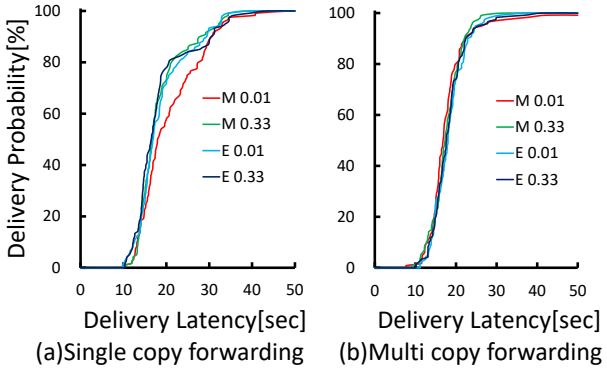


Figure 4: Packet delivery latency for SCF and MCF. We evaluated with Manhattan(M) and Euclidean(E) distance, and $\alpha = 0.01$ and $\alpha = 0.33$.

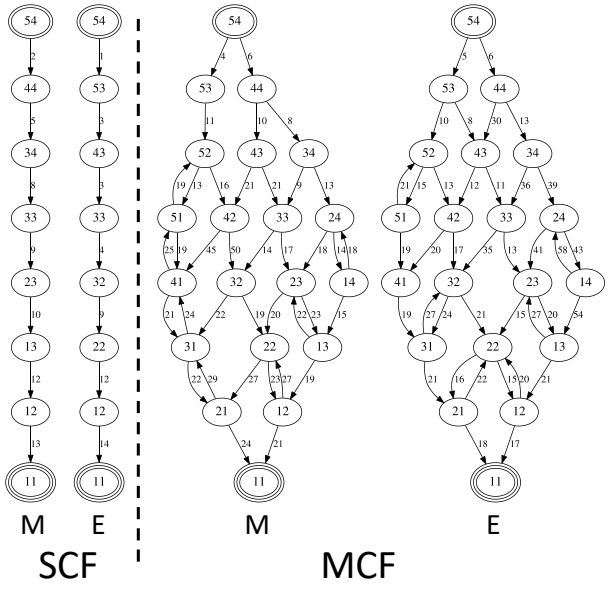


Figure 5: 4 samples of packet delivery paths for SCF and MCF. These samples were taken with Manhattan(M) and Euclidean(E) distance and $\alpha = 0.33$. The arrow label shows the time from the packet generation at node (5,4).

4.3 Delivery Latency

Fig.4 shows the cumulative distribution function of the delivery latency for all the cases. In general, MCF performed better compared to SCF, especially for the last 20% of slow-delivery packets. In SCF, Manhattan+$\alpha = 0.33$ relatively performed well for all the latency situations. There are no clear differences in MCF. This is probably

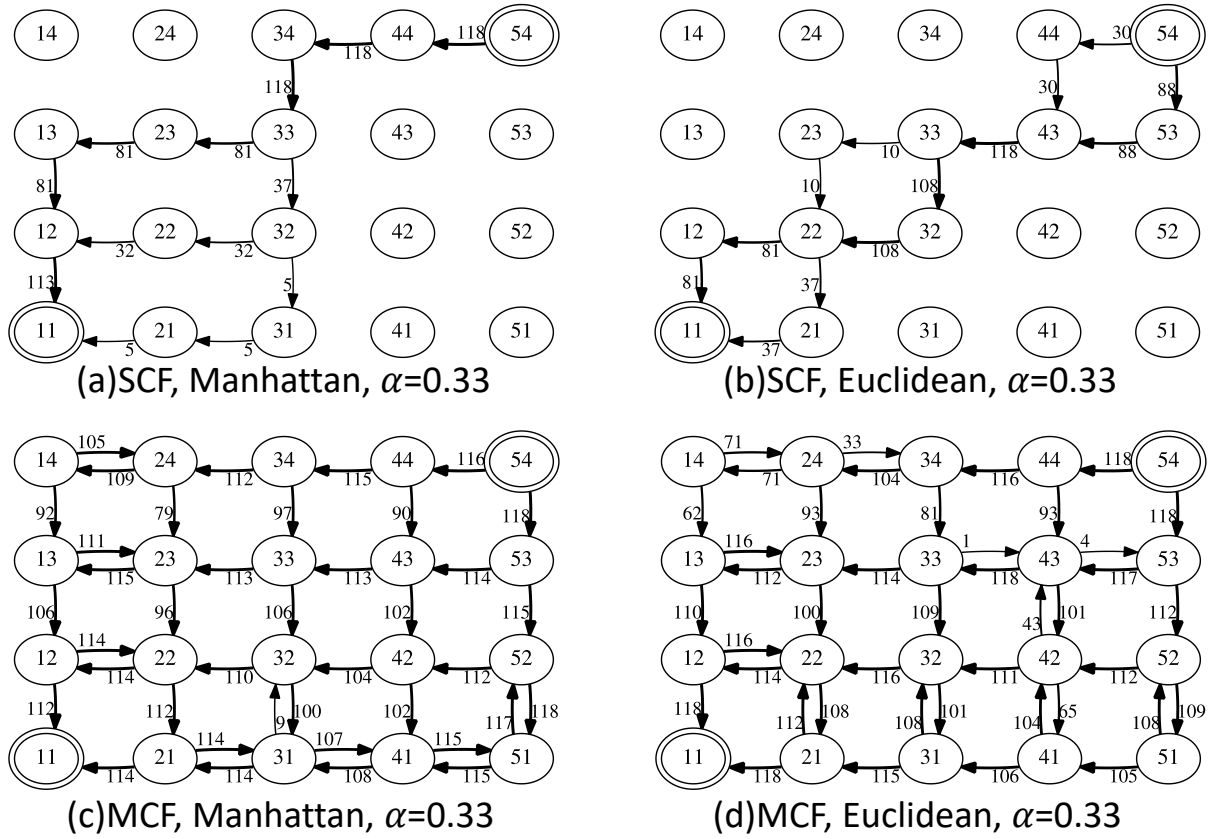


Figure 6: Aggregated delivery paths from (5,4) to (1,1). The arrow label shows the number of packets successfully forwarded on this link.

because, in MCF as described in section 4.4, almost all the nodes got the copies of all the packets – the distance metrics and α did not practically make a difference for determining $\mathbf{next}_{\mathrm{MCF}}(\mathbf{t}, p)$.

4.4 Packet Delivery Path

Fig.5 shows 4 samples of packet delivery paths, i.e., transmission traces, for SCF and MCF with Manhattan distance and Euclidean distance, and $\alpha = 0.33$. This visualization is based on the receival of $\mathrm{F_r}$equest. The label on the arrow indicates when this transmission has been made (in second) from the generation of the packet. We can observe that (1) packet delivery with SCF is straight-forward – directly sending the packet to the destination with the minimum copies, and that (2) packet delivery with MCF has made too many copies in the network – distributing to all the nodes sometimes causing backward-forwarding. The average number of copies made in the experiments are 7.00 for SCF and 18.4 for MCF.

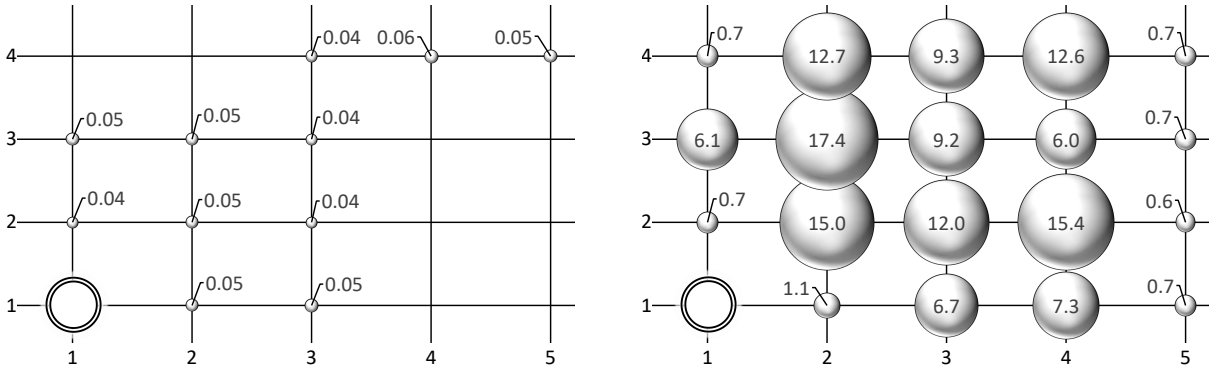
Fig.6 shows the aggregated delivery paths for all the packets, i.e., 118 packets generated at node (5,4). The label shows the number of packets successfully forwarded between the two nodes – duplicated transmissions on the same link are not counted. From the these graphs, we can observe that in SCF, Manhattan distance has allowed the packets to take the route based on the cost of the links, and that Euclidean distance has concentrated the packets around the diagonal paths e.g., (2,2) and (3,3). In MCF, we cannot observe the differences between Manhattan and Euclidean as they copied the packets to basically all the nodes.

4.5 Buffer Occupancy

Fig.7 shows the average buffer occupancy, i.e., the average number of packets contained in the packet buffer, during the experiment for SCF and MCF. There are clear differences between them. In SCF, the average buffer occupancy were around 0.05 packets, but in MCF, we observed much larger occupancies and variance among the nodes. These results indicates that in SCF, a very few number of packets have been remained in the network, where as, in MCF, many packets have been remained in the network in order to make re-transmission of packets to the next hop nodes.

5 DISCUSSION

We have proposed the architecture and routing schemes of IR-XY-PV. During the discussion, we have identified several design choices such as single-copy forwarding (SCF) and multi-copy forwarding (MCF), Manhattan distance and Euclidean distance, and α and other parameters. With the 20-node scale experiment, we have studied the features of these design choices regarding to packet delivery success rate, delivery latency, paths, and buffer occupancy.



(a)SCF, Manhattan, α=0.33 (b)MCF, Manhattan, α=0.33

Figure 7: Average occupancies of packet buffers for SCF and MCF with Manhattan distance and $\alpha = 0.33$. The destination (1,1), which did not use the packet buffer, is specially marked.

According to the result, MCF performed better than SCF when we just consider the latency for one packet delivery. However, MCF consumed much larger buffer spaces and transmissions. MCF may also potentially lose packet by deleting all the propagated packet copies by the intensional deletion scheme defined in section 3.3.2 – which has actually happened in the experiment.

In the future, we need to consider the case where all the nodes generate packets to a sink node, i.e., to the same destination. Congestion in transmissions and buffer overflow will probably happen especially in MCF – causing slower delivery and larger packet loss than SCF. However we cannot simply use SCF, because SCF also has a chance of packet loss. Thus, we consider that more study is necessary for MCF algorithm.

6 CONCLUSION

In this paper, we have proposed the architecture and routing schemes of an infrared multihop communication for XY-coordinated PV modules (IR-XY-PV). We proposed "XY routing" with several design choices on (1) IR transceivers, (2) neighbor discovery algorithm, (3) disruption tolerant forwarding, (4) single-copy and multi-copy, (4) Manhattan distance and Euclid distance, (5) link cost management and other configuration parameters. We have developed 20 node scale IR-XY-PV testbed network and studied the delivery success rate, delivery latency, delivery paths, and memory usage for those several design choices. With the experiment, we confirmed that IR-XY-PV can provide practically enough performance with our proposed architecture and routing scheme.

REFERENCES

[1] F. Cadger, K. Curran, J. Santos, and S. Moffett. A survey of geographical routing in wireless ad-hoc networks. *IEEE Communications Surveys & Tutorials*, 15(2):621–653, 2013.
[2] Y. Cao and Z. Sun. Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges. *IEEE Communications surveys & tutorials*, 15(2):654–677, 2013.
[3] C.-H. Chang, J.-J. Zhu, and H.-L. Tsai. Model-based performance diagnosis for PV systems. In *IEEE SICE 2010*, pages 2139–2145, 2010.
[4] P.-C. Cheng, K. C. Lee, M. Gerla, and J. Härri. GeoDTN+ Nav: geographic DTN routing with navigator prediction for urban vehicular environments. *Mobile Networks and Applications*, 15(1):61–82, 2010.
[5] H. Elgala, R. Mesleh, and H. Haas. Indoor optical wireless communication: potential and state-of-the-art. *IEEE Communications Magazine*, 49(9), 2011.
[6] N. Forero, J. Hernández, and G. Gordillo. Development of a monitoring system for a PV solar plant. *Energy Conversion and Management*, 47(15):2329–2336, 2006.
[7] J. Han, I. Lee, and S.-H. Kim. User-friendly monitoring system for residential PV system based on low-cost power line communication. *IEEE Transactions on Consumer Electronics*, 61(2):175–180, 2015.
[8] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE network*, 15(6):30–39, 2001.
[9] H. Ochiai and H. Ikegami. PPLC-PV: A pulse power line communication for series-connected PV monitoring. In *IEEE SmartGridComm*, pages 338–344, 2016.
[10] P. Papageorgas, D. Piromalis, K. Antonakoglou, G. Vokas, D. Tseles, and K. Arvanitis. Smart solar panels: In-situ monitoring of photovoltaic panels based on wired and wireless sensor networks. *Energy Procedia*, 36:535–545, 2013.
[11] C. Ranhotigamage and S. C. Mukhopadhyay. Field trials and performance monitoring of distributed solar panels using a low-cost wireless sensors network for domestic applications. *IEEE Sensors Journal*, 11(10):2583–2590, 2011.
[12] F. J. Sanchez-Pacheco, P. J. Sotorrio-Ruiz, J. R. Heredia-Larrubia, F. Perez-Hidalgo, and M. Sidrach de Cardona. PLC-based PV plants smart monitoring system: field measurements and uncertainty estimation. *IEEE Transactions on Instrumentation and Measurement*, 63(9):2215–2222, 2014.
[13] G. M. Tina and A. D. Grasso. Remote monitoring system for stand-alone photovoltaic power plants: The case study of a PV-powered outdoor refrigerator. *Energy Conversion and Management*, 78:862–871, 2014.
[14] L. Zhao, F. Li, and Y. Wang. Hybrid position-based and DTN forwarding in vehicular ad hoc networks. In *IEEE VTC*, pages 1–5, 2012.