DAFTAR PUSTAKA

Asaad, S. M., and Maghdid, H. S., (2022), A Comprehensive Review of Indoor/Outdoor Localization Solutions in IoT era: Research Challenges and Future Perspectives, Computer Networks, vol. 212, p. 109041, doi: 10.1016/J.COMNET.2022.109041

Ahmed, A. U., Bergmann, N. W., Arablouei, R., Kusy, B., Hoog, F. De , & R. Jurdak, (2018), Poster Abstract- Fast Indoor Localization Using WiFi Channel State Information, 2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks

Archer, D. H., McInturff, K., Mintzer, A. I., Theis, W. H., (1993) Broadband Direction Finding, US Patent, 5,274,389,

Balanis, C. A., Ioannides, P.I., (2007a), Intoduction to Smart Antennas. Morgan & Claypool

Balanis, C. A., (2016b), Antenna Theory Analysis and Design, Fourth Edition. John Wiley & Sons, Inc.

Balogh, L., Kollar, I., (2003), Angle of arrival estimation based on interferometer principle. IEEE International Symposium on Intelligent Signal Processing. 219 – 223

Brandao, A. L., Sydor, J., Brett W., Scott, J., Joe, P., and Hung, D., (2005), "5GHz RLAN Interference on Active Meteorological Radars", IEEE 61st Vehicular Technology Conference

He, C., Cao, A., Chen, J., Liang, X., Zhu, W., Geng, J., Jin, R., (2018), Direction Finding by Time-Modulated Linear Array, in IEEE Transactions on Antennas and Propagation, vol. 66, no. 7, pp. 3642-3652, doi: 10.1109/TAP.2018.2835164.

Chen, Z., Zhang, Y., (2018), Monostatic Multi-Source Direction Finding Based on I/Q Radio Frequency Data". AEU - International Journal of Electronics and Communications

Chong, Z., HongYang, Y., (2013), Indoor matching location algorithm based on WIFI signal strength, Research Institute of Electronic Science and Technology. University of Electronic Science and Technology of China Chengdu. China

Choi, J., Lee, G., Choi, S., and Bahk, S., (2022), Smartphone Based Indoor Path Estimation and Localization Without Human Intervention, in IEEE Transactions on Mobile Computing, vol. 21, no. 2, 681-695, doi: 10.1109/TMC.2020.3013113

Cordill, B. D., Seguin, S. A., Cohen, L., (2013), Electromagnetic interference to radar receivers due to in-band OFDM communications systems, 2013 IEEE International Symposium on Electromagnetic Compatibility. 72 – 75

Ding, W., Zhong, Q., Wang, Y., Guan, C., and Fang, B., (2022), Target Localization in Wireless Sensor Networks Based on Received Signal Strength and Convex Relaxation, Sensors, vol. 22, no. 3, 733, doi: 10.3390/s22030733

Duraj, D., Rzymowski, M., Nyka, K., and Kulas, L., (2020), RSS-Based DoA Estimation Using ESPAR Antenna for V2X Applications in 802.11p Frequency Band, 2020 14th European Conference on Antennas and Propagation (EuCAP), 1-5, doi: 10.23919/EuCAP48036.2020.9135688

Dolinska, I., Jakubowski, M., Masiukiewicz, A., (2017), Interference comparison in Wi-Fi 2.4 GHz and 5 GHz bands, 2017 International Conference on Information and Digital Technologies (IDT). 106 – 112

Duplouy, J. (2019), Wideband Reconfigurable Vector Antenna for 3-D Direction Finding Application. Electromagnetism, PhD. Thesis, Institut National Polytechnique de Toulouse - INPT

Electronic Communications Commitee. (2014), ECC Report 192 The Current Status of DFS (Dynamic Frequency Selection) In the 5 GHz Frequency Range

Floch, J-J., Antenna Beamforming for Tracking a transmitter signal, US Patent, US 9,612,3372 B2, 4 April 2017

Gyoda. K., and Ohira, T., (2000), Design of electronically steerable passive array radiator (ESPAR) antennas," IEEE Antennas and Propagation Society International Symposium. Transmitting Waves of Progress to the Next Millennium. 2000 Digest. Held in conjunction with: USNC/URSI National Radio Science Meeting (C, Salt Lake City, UT, USA, 2000, vol.2, 922-925, doi: 10.1109/APS.2000.875370

Harrison, R. W. S., and Jessup, M. (2012), A novel Log periodic implementation of a 700 MHz – 6 GHz slant polarised fixed-beam antenna array for direction finding applications". 2012 42nd European Microwave Conference.

Herrick, D. L., Vernon, M., Matthews, W. F. III, Hudson, (1997), Apparatus and methode for finding a signal emission source, US Patent, 5,625,364

Hidayat, A., Palantei, E., Syarif, S., Irawadi, D., Munawar, STA., (2019), Alat Pencari Arah Pemancar Menggunakan Antena Enam Elemen Antena dan Monitoringnya Secara Online, Direktorat Jenderal Kekayaan Intelektual Kementerian Hukum dan Hak Asasi Manusia Republik Indonesia, Int.Cl.2017.01/H 01Q 3/00(2006.01)

Holleman, J., Michelson, D., Galli, G., Germann, U., and Peura, M., (2006), Quality

Information fo Radars and Radar Data. OPERA 2005 19

Jian, H. X., Hao, W. (2017), WIFI Indoor Location Optimization Method Based on Position Fingerprint Algorithm, 2017 International Conference on Smart Grid and Electrical Automation (ICSGEA). 585 – 588

Kementerian Komunikasi dan Informatika RI. (2021). Data Statistik Direktorat Jenderal Sumber Daya dan Perangkat Pos dan Informatika 2020, Bogor: IPB Press

Khasanova, A. M., (2021), Detection of Attacks on Wi-Fi Access Points, 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). 28 – 31

Kulas, L., (2017), Simple 2-D Direction-of-Arrival Estimation Using an ESPAR Antenna, IEEE Antennas and Wireless Propagation Letters, vol. 16, 2513-2516, doi: 10.1109/LAWP.2017.2728322

Kwapisiewicz, P., Groth, M., Rzymowski M., and Kulas, L., (2019), "SDR-Based DoA Estimation Using ESPAR Antennas with Simplified Beam Steering," 2019 13th European Conference on Antennas and Propagation (EuCAP), pp. 1-4

Lee, Y.-H., Lin, C.-S., (2016), WiFi Fingerprinting for Indoor Room Localization Based on CRF Prediction. 2016 International Symposium on Computer, Consumer and Control

Lim, C.-H., Wan, Y., Ng, B.-P., and See, C.-M., (2007), A Real-Time Indoor WiFi Localization System Utilizing Smart Antennas. IEEE Transactions on Consumer Electronics. vol. 53. no. 2

Mainsuri, Palantei, E., Areni, I. S., Wardi, Baharuddin, M., Dewiani, Sunarno, Setijadi, E., Hidayat, A., (2020), A 923 Mhz Steerable Antenna for Low Power Wide Area Network (LPWAN), 2020 IEEE International Conference on Communication, Networks and Satellite (Comnetsat), 246 - 250

Niu, J., Wang, B., Cheng, L., Rodrigues, J. J. P. C., (2015), WicLoc- An indoor localization system based on WiFi fingerprints and crowdsourcing, IEEE ICC 2015 - Mobile and Wireless Networking Symposium

Ohira, T., and Gyoda, K., (2001), Hand-held microwave direction-of-arrival finder based on varactor-tuned analog aerial beamforming, APMC 2001. 2001 Asia-Pacific Microwave Conference (Cat. No.01TH8577), vol.2, 585-588, doi: 10.1109/APMC.2001.985441

Ouyang, W., and Gong, X., (2019), An Electronically Steerable Parasitic Array Radiator (ESPAR) Using Cavity-Backed Slot Antennas, in IEEE Antennas and Wireless Propagation Letters, vol. 18, no. 4, 757-761, doi:

10.1109/LAWP.2019.2902037

Palantei, E., Thiel, D.V., (2007), The Impact of Bias Voltage on the Performance of a P.I.N. Diode Loaded Smart Antenna, Journal of the Japan Soceity of Applied Electromagnetics and Mechanics Vol. 15. No. 3, 274-277

Peraturan Menteri Komunikasi dan Informatika Nomor 2 Tahun 2023 tentang Penggunaan Spektrum Frekuensi Radio Berdasarkan Izin Kelas, (https://jdih.kominfo.go.id/storage/files/1683195383-Salinan_PM_Kominfo_Nomor_2_Tahun_2023_Tanpa_Paraf_TTE.pdf, diakses 15 Juli 2023)

Rasheed, I., Schiller, C. T., (2014), Automatic high resolution adaptive beam-steering, US Patent, US 2014/0266894 A1

Saltikof, E., Cho, J. Y. N., Tristant, P. A., Huuskonen, Allmon L., Cook, R., et al., (2016). The Threat To Weather Radars By Wireless Technology. Bulletin of the American Meteorological Society

Sanam, T. F., and Godrich, H., (2018), An Improved CSI Based Device Free Indoor Localization Using Machine Learning Based Classification Approach," 2018 26th European Signal Processing Conference (EUSIPCO), 2390-2394, doi: 10.23919/EUSIPCO.2018.8553394

Shih, Y. -Y., Pang A. -C., and Hsiu, P. -C., (2018), A Doppler Effect Based Framework for Wi-Fi Signal Tracking in Search and Rescue Operations," in IEEE Transactions on Vehicular Technology, vol. 67, no. 5, 3924-3936, doi: 10.1109/TVT.2017.2752766

Ssekidde, P., Steven Eyobu, O., Han, D. S., and Oyana, T. J., (2021), Augmented CWT Features for Deep Learning-Based Indoor Localization Using WiFi RSSI Data," Applied Sciences, vol. 11, no. 4, 1806, , doi: 10.3390/app11041806

Therese, A., Babu, M. N., and Krishna, D. P. M., (2018), Passive Localisation Based On Correlative Interferometry, Procedia Computer Science. Vol.143. 2 – 9

Tian, Z., Wang, Z., Li, Z., and Zhou, M., (2019), RTIL- A Real-Time Indoor Localization System by Using Angle of Arrival of Commodity WiFi Signal. Chongqing University of Posts and Telecommunications, Chongqing. China.

Tuncer, T. E., Friedlander, B., (2009), Classical and Modern Direction-of-Arrival Estimation. Academic Press Publications

Tzur, A., Amrani, O., Wool, A., (2015), Direction Finding of rogue Wi-Fi access points using an off-the-shelf MIMO–OFDM receiver. School of Electrical Engineering, Tel-Aviv University, Israel

Wang, X., Gao, L., Mau, S., and Pandey, S., (2017), CSI-Based Fingerprinting for Indoor Localization: A Deep Learning Approach," in IEEE Transactions on Vehicular Technology, vol. 66, no. 1, 763-776, doi: 10.1109/TVT.2016.2545523

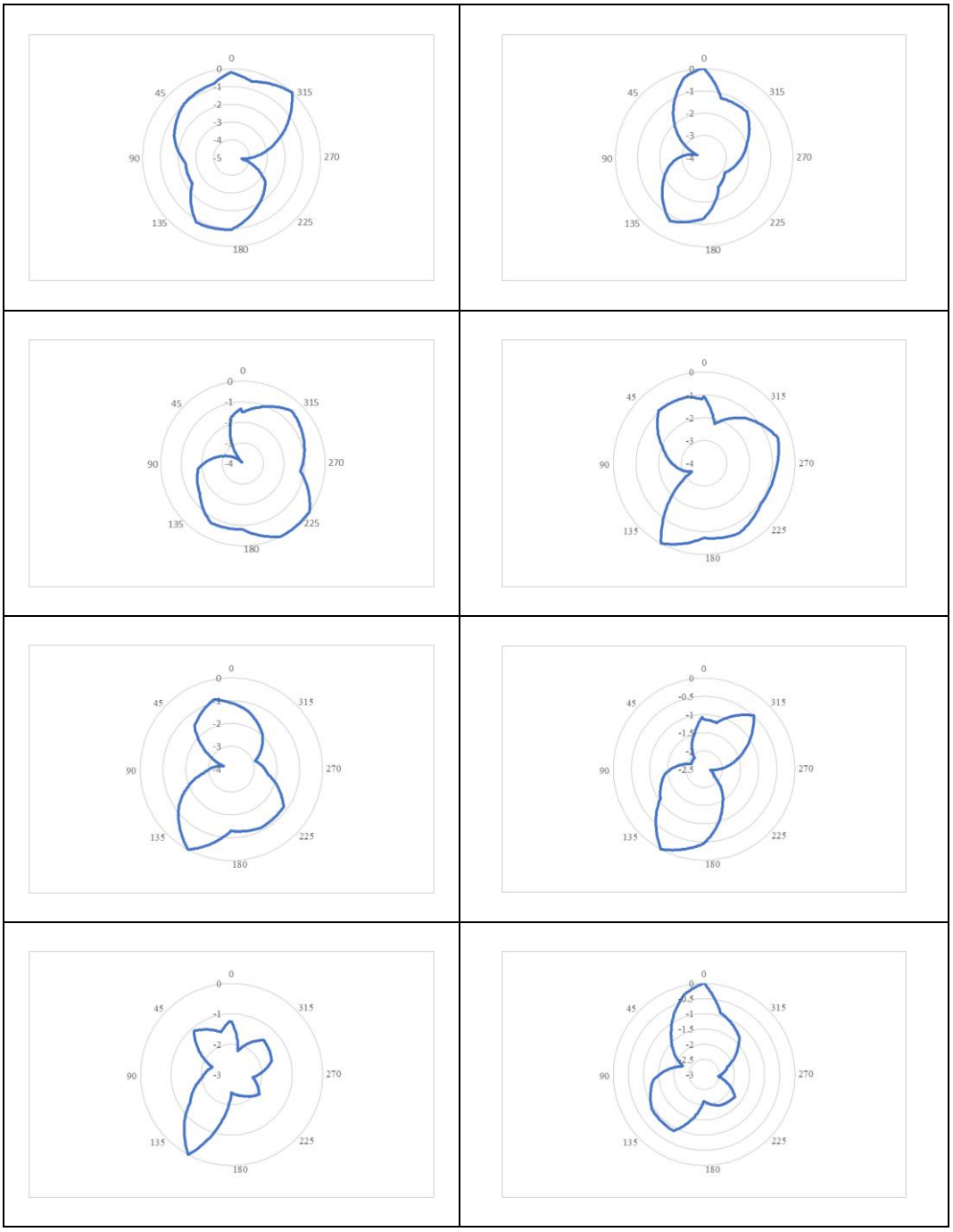
Won, H., Hong, Y.-K., Isbell, K., Vanderburgh, L., Platt, J., and Choi, M., (2019), Evaluation on Pseudo-Doppler Antenna Array using Software-Defined-Radio, 2019 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting, 1835 – 1836

Xue, J., Liu, J., Sheng, M., Shi, Y., and Li, J., (2020), A WiFi fingerprint based high-adaptability indoor localization via machine learning, China Communications, vol. 17, no. 7, 247-259, doi: 10.23919/J.CC.2020.07.018.

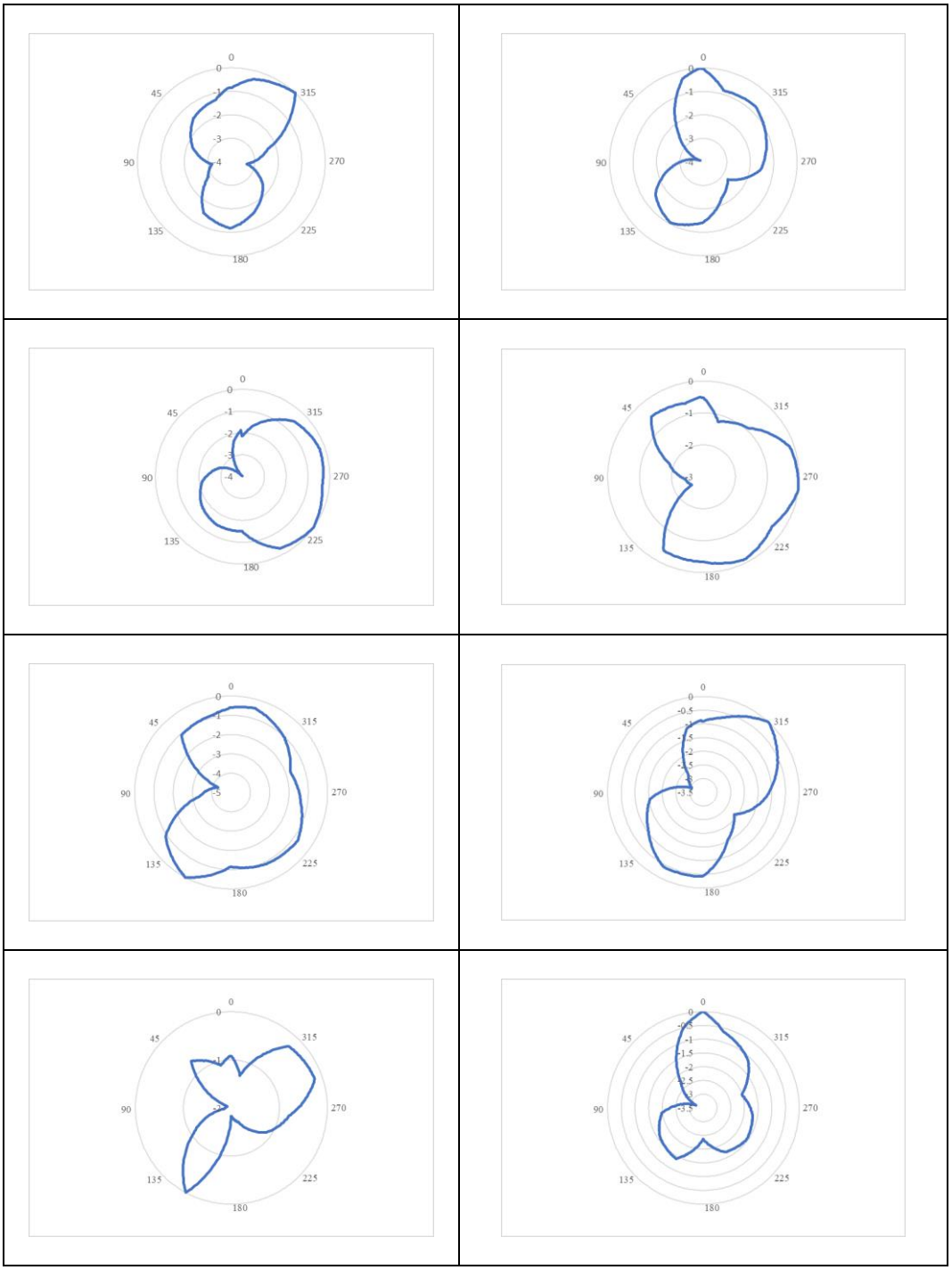
LAMPIRAN

Lampiran 1. Pola radiasi antena hasil pengukuran di *anechoic chamber*

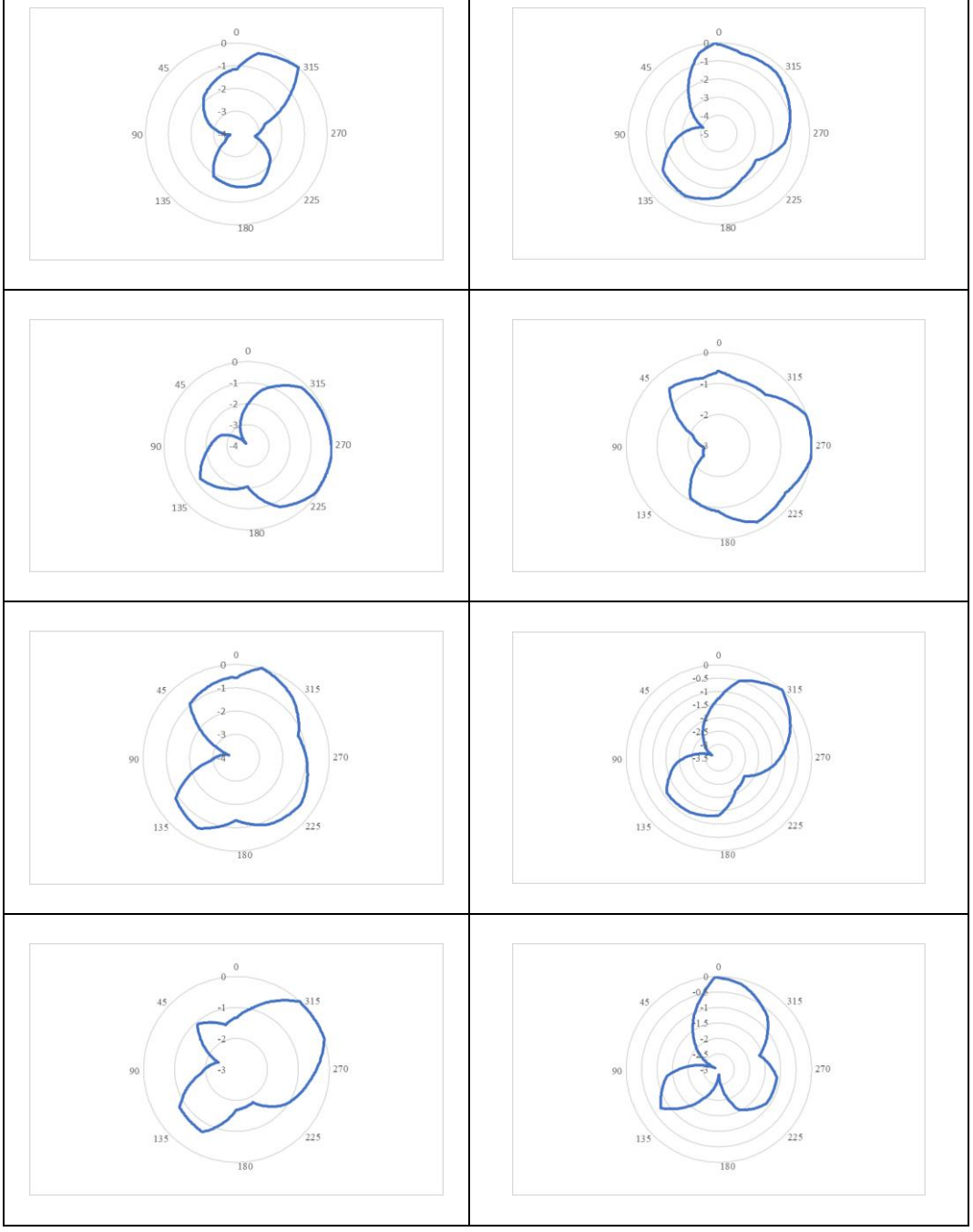
Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S1O7 dan *beam steering* pada frekuensi 5,5 GHz



Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S1O7 dan *beam steering* pada frekuensi 5,56 GHz

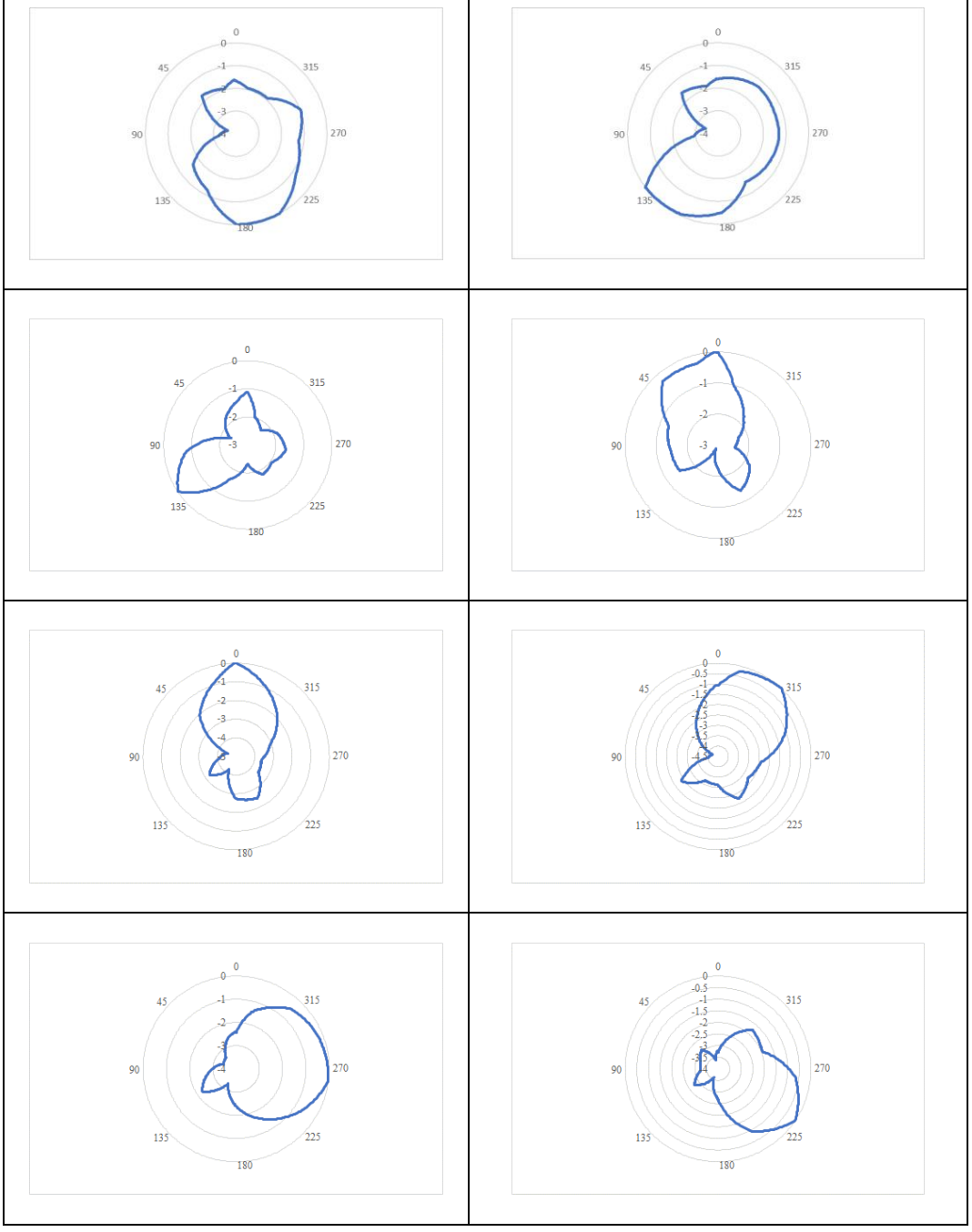


Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S1O7 dan *beam steering* pada frekuensi 5. 6 GHz

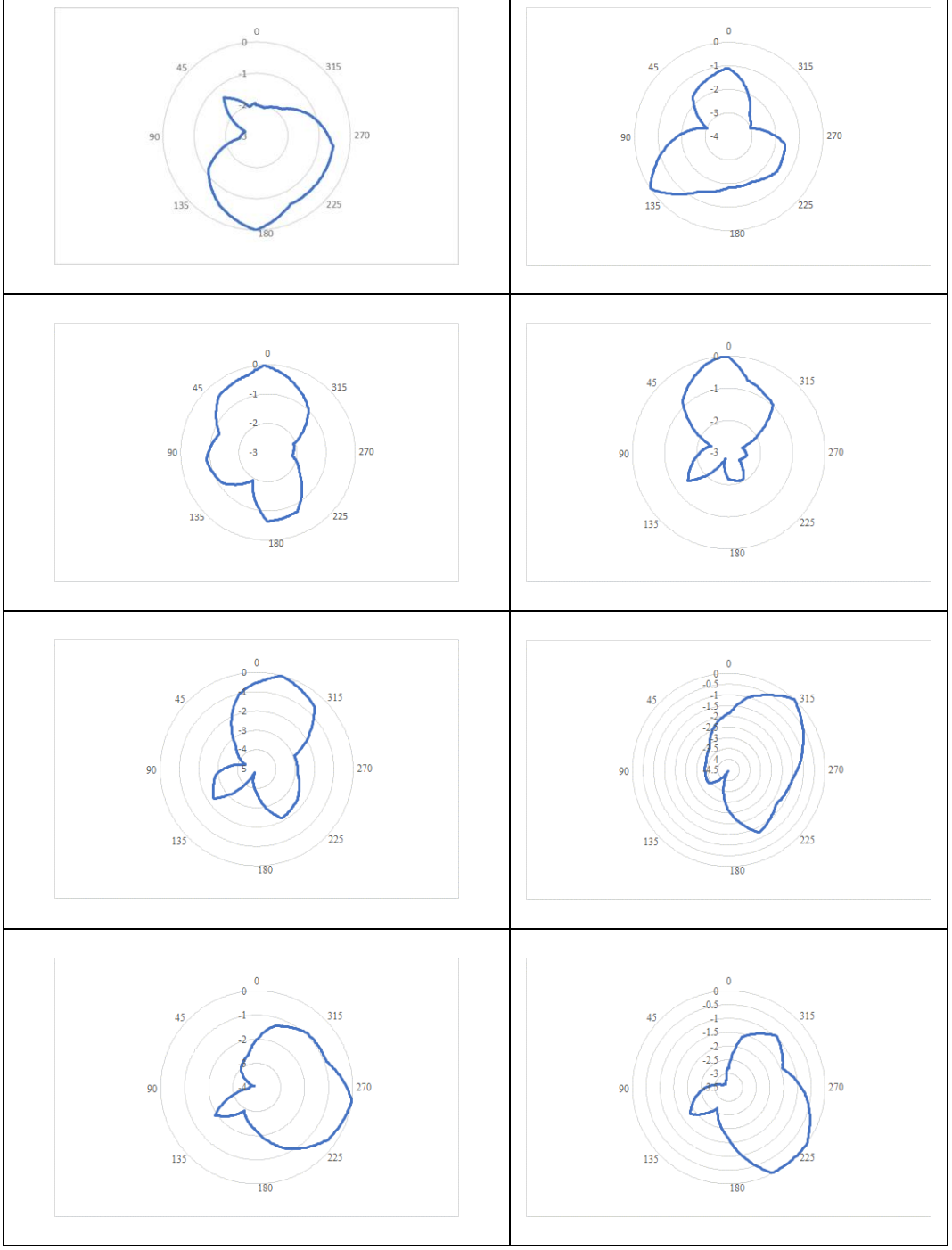


Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S2O6 dan *beam steering* pada frekuensi 5,56 GHz

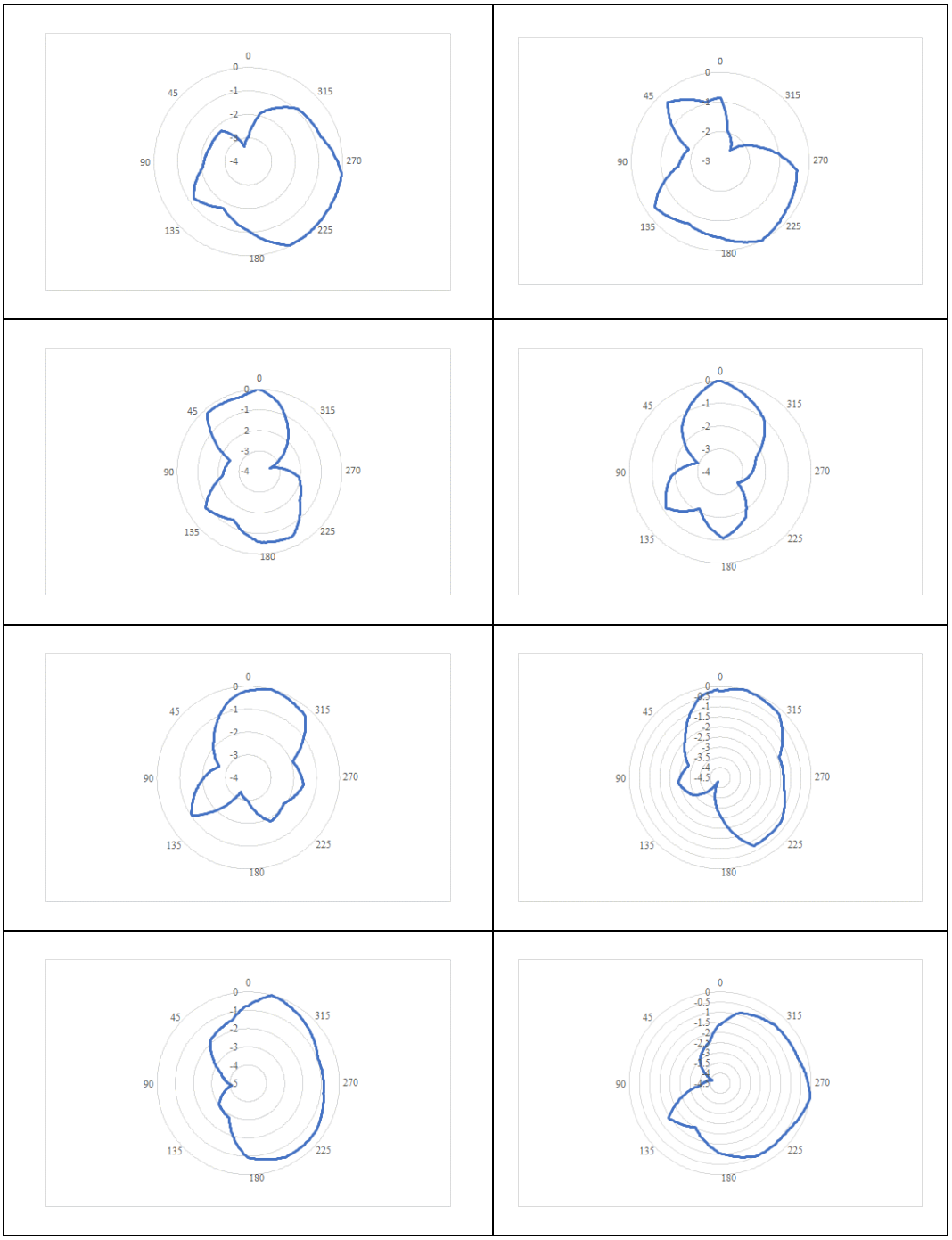
Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S3O5 dan *beam steering* pada frekuensi 5,56 GHz



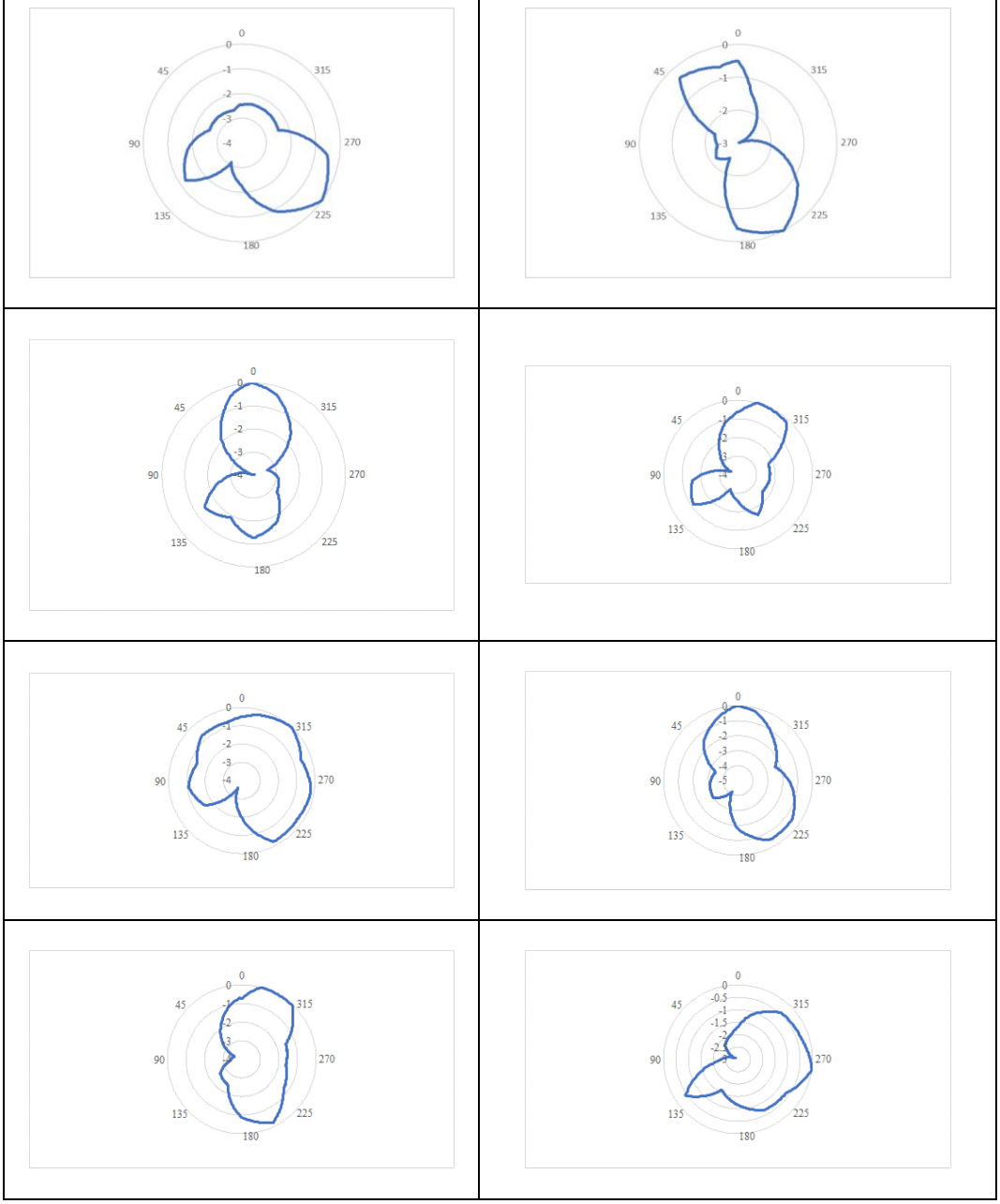
Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S4O4 dan *beam steering* pada frekuensi 5,56 GHz



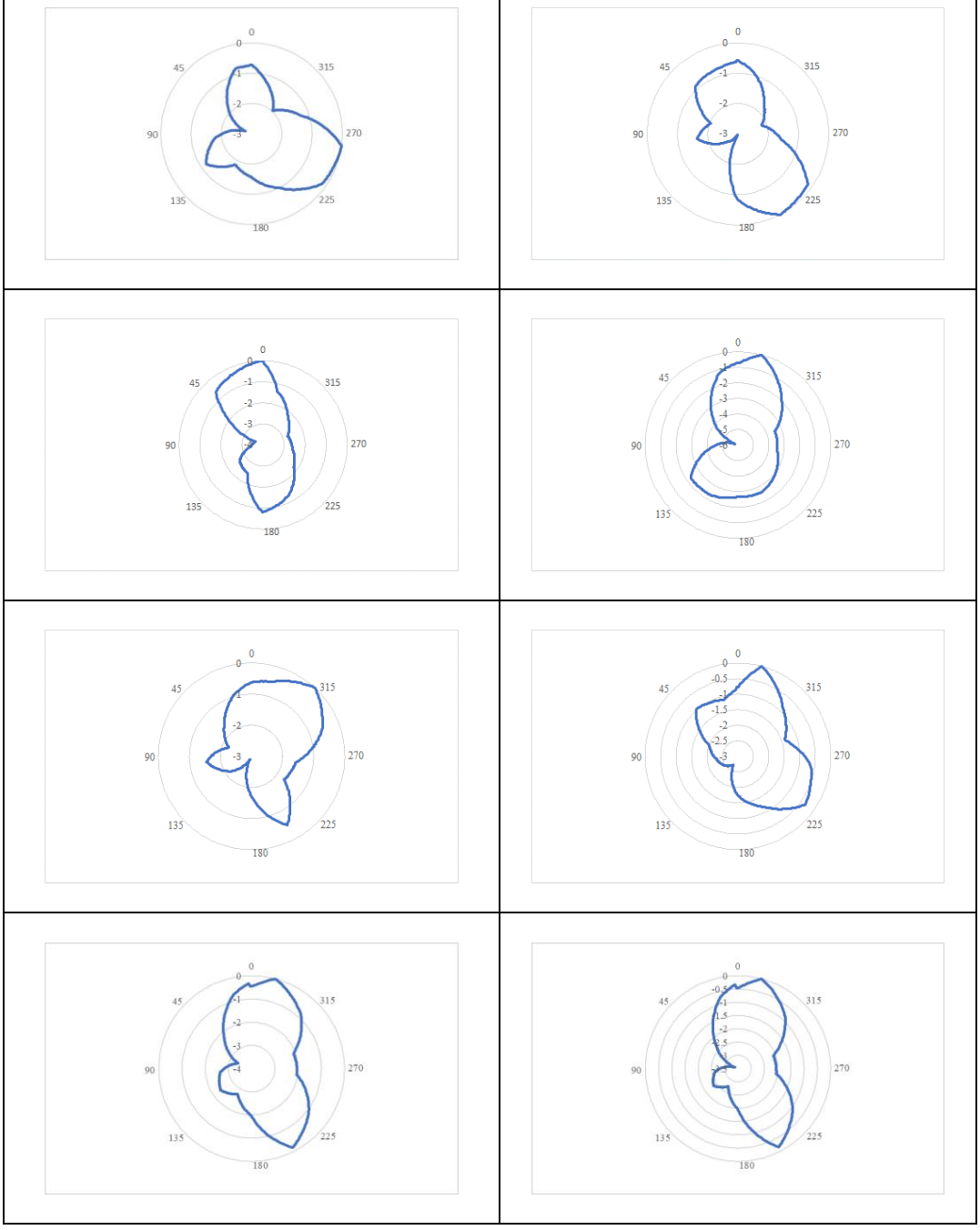
Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S5O3 dan *beam steering* pada frekuensi 5,56 GHz



Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S6O2 dan *beam steering* pada frekuensi 5,56 GHz



Normalisasi hasil pengukuran pola radiasi antena frekuensi kombinasi S7O1 dan *beam steering* pada frekuensi 5,56 GHz



Lampiran 2. Program

Program kendali antena untuk kalibrasi

```
import os
import sys
import subprocess
import errno

import RPi.GPIO as GPIO

SW1 = 11
SW2 = 13
SW3 = 15
SW4 = 19
SW5 = 21
SW6 = 23
SW7 = 29
SW8 = 31

def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(SW1,GPIO.OUT)
    GPIO.setup(SW2,GPIO.OUT)
    GPIO.setup(SW3,GPIO.OUT)
    GPIO.setup(SW4,GPIO.OUT)
    GPIO.setup(SW5,GPIO.OUT)
    GPIO.setup(SW6,GPIO.OUT)
    GPIO.setup(SW7,GPIO.OUT)
    GPIO.setup(SW8,GPIO.OUT)

def S2O61():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 1')


def S2O62():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 2')


def S2O63():
    GPIO.output(SW1,GPIO.LOW)
```

```
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S2O6..Arah 3')


def S2O64():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S2O6..Arah 4')


def S2O65():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 5')


def S2O66():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 6')


def S2O67():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 7')
```

```
def S2O68():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 8')


def S3O51():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 1')

def S3O52():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 2')

def S3O53():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 3')

def S3O54():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
```

```
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 4')


def S3O55():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 5')


def S3O56():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 6')


def S3O57():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 7')


def S3O58():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 8')
```

```
def S4O41():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 1')


def S4O42():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 2')


def S4O43():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 3')


def S4O44():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 4')


def S4O45():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
```

```
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 5')


def S4O46():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 6')


def S4O47():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 7')


def S4O48():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 8')


def S5O31():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 1')


def S5O32():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
```

```
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 2')


def S5O33():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 3')


def S5O34():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 4')


def S5O35():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 5')


def S5O36():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 6')
```

```
def S5O37():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 7')


def S5O38():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 8')

def S6O21():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 1')

def S6O22():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 2')

def S6O23():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 3')
```

```
def S6O24():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 4')

def S6O25():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 5')

def S6O26():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 6')

def S6O27():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S6O2..Arah 7')

def S6O28():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S6O2..Arah 8')
```

```
def S1():
    print ('SW1 ON..')
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)


def S2():
    print ('SW2 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S3():
    print ('SW3 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S4():
    print ('SW4 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S5():
    print ('SW5 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
```

```
def S6():
    print ('SW6 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S7():
    print ('SW7 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)

def S8():
    print ('SW8 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)



def out_pin():
    SW1_out = GPIO.input(SW1)
    SW2_out = GPIO.input(SW2)
    SW3_out = GPIO.input(SW3)
    SW4_out = GPIO.input(SW4)
    SW5_out = GPIO.input(SW5)
    SW6_out = GPIO.input(SW6)
    SW7_out = GPIO.input(SW7)
    SW8_out = GPIO.input(SW8)

    print("SW1 : ", SW1_out)
    print("SW2 : ", SW2_out)
    print("SW3 : ", SW3_out)
    print("SW4 : ", SW4_out)
    print("SW5 : ", SW5_out)
    print("SW6 : ", SW6_out)
    print("SW7 : ", SW7_out)
    print("SW8 : ", SW8_out)
```

```
def main():
    print("Contoh konfigurasi antena : ")
    print("S5031 = 5 elemen dishort dan 3 elemen open pada arah ke
1 ")
    argument = input("Masukkan konfigurasi antena : ")
    argument = argument+"()"
    exec(argument)
    out_pin()


if __name__ == '__main__':
    GPIO.cleanup()
    setup()
    while(True):
        try:
            main()

        # When 'Ctrl+C' is pressed, the program destroy() will be
executed.
        except KeyboardInterrupt:
            GPIO.cleanup()
            False
            exit
```

Program pengukuran pada titik stasioner

```
import os
import sys
import subprocess
import errno
import math
import RPi.GPIO as GPIO
import csv
import os
from time import sleep
import datetime
import numpy as np



# Mendapatkan tanggal dan waktu saat ini
now = datetime.datetime.now()
date_time = now.strftime("%m%d%H%M")

SW1 = 11
SW2 = 13
SW3 = 15
SW4 = 19
SW5 = 21
SW6 = 23
SW7 = 29
SW8 = 31

def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(SW1,GPIO.OUT)
    GPIO.setup(SW2,GPIO.OUT)
    GPIO.setup(SW3,GPIO.OUT)
    GPIO.setup(SW4,GPIO.OUT)
    GPIO.setup(SW5,GPIO.OUT)
    GPIO.setup(SW6,GPIO.OUT)
    GPIO.setup(SW7,GPIO.OUT)
    GPIO.setup(SW8,GPIO.OUT)

def S2O61():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 1')


def S2O62():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
```

```
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 2')


def S2O63():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S2O6..Arah 3')


def S2O64():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S2O6..Arah 4')


def S2O65():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 5')


def S2O66():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 6')
```

```
def S2O67():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 7')


def S2O68():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 8')


def S3O51():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 1')


def S3O52():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 2')


def S3O53():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
```

```
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 3')


def S3O54():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 4')


def S3O55():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 5')


def S3O56():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 6')


def S3O57():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 7')


def S3O58():
    GPIO.output(SW1,GPIO.LOW)
```

```
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 8')


def S4O41():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 1')


def S4O42():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 2')


def S4O43():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 3')


def S4O44():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
```

```
    print('S4O4..Arah 4')


def S4O45():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 5')


def S4O46():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 6')


def S4O47():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 7')


def S4O48():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 8')


def S5O31():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
```

```
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 1')


def S5O32():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 2')


def S5O33():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 3')


def S5O34():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 4')


def S5O35():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 5')


def S5O36():
```

```
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 6')


def S5O37():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 7')


def S5O38():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 8')

def S6O21():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 1')

def S6O22():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 2')
```

```
def S6O23():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 3')

def S6O24():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 4')

def S6O25():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 5')

def S6O26():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 6')

def S6O27():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S6O2..Arah 7')

def S6O28():
```

```
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S6O2..Arah 8')

def S7O11():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 1')

def S7O12():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 2')

def S7O13():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 3')

def S7O14():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 4')


def S7O15():
```

```
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 5')

def S7O16():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 6')

def S7O17():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 7')

def S7O18():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S7O1..Arah 8')

def S8O0():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S8O0...')

def S0O8():
    GPIO.output(SW1,GPIO.LOW)
```

```
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S0O8...')



def S1O71():
    print ('SW1 ON..')
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)


def S1O72():
    print ('SW2 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O73():
    print ('SW3 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O74():
    print ('SW4 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
```

```
    GPIO.output(SW8,GPIO.LOW)

def S1O75():
    print ('SW5 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O76():
    print ('SW6 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O77():
    print ('SW7 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)

def S1O78():
    print ('SW8 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)



def out_pin():
    SW1_out = GPIO.input(SW1)
    SW2_out = GPIO.input(SW2)
    SW3_out = GPIO.input(SW3)
    SW4_out = GPIO.input(SW4)
    SW5_out = GPIO.input(SW5)
    SW6_out = GPIO.input(SW6)
    SW7_out = GPIO.input(SW7)
```

```
    SW8_out = GPIO.input(SW8)

    print("SW1 : ", SW1_out)
    print("SW2 : ", SW2_out)
    print("SW3 : ", SW3_out)
    print("SW4 : ", SW4_out)
    print("SW5 : ", SW5_out)
    print("SW6 : ", SW6_out)
    print("SW7 : ", SW7_out)
    print("SW8 : ", SW8_out)



def main():
    bearing = input("Masukkan arah antena : ")
    bearing = int(bearing)
    bearing = 360-bearing

    input_dir = '/home/rasp/Tesis'
    output_file = str(bearing)+'_static_'+date_time+'.csv'

    directory = '/home/rasp/Tesis/'
    command =f"iw dev wlan0 station dump | grep signal"



    subprocesses = [
        'S1O71', 'S1O72', 'S1O73', 'S1O74', 'S1O75', 'S1O76',
'S1O77', 'S1O78',
        'S2O61', 'S2O62', 'S2O63', 'S2O64', 'S2O65', 'S2O66',
'S2O67', 'S2O68',
        'S3O51', 'S3O52', 'S3O53', 'S3O54', 'S3O55', 'S3O56',
'S3O57', 'S3O58',
        'S4O41', 'S4O42', 'S4O43', 'S4O44', 'S4O45', 'S4O46',
'S4O47', 'S4O48',
        'S5O31', 'S5O32', 'S5O33', 'S5O34', 'S5O35', 'S5O36',
'S5O37', 'S5O38',
        'S6O21', 'S6O22', 'S6O23', 'S6O24', 'S6O25', 'S6O26',
'S6O27', 'S6O28',
        'S7O11', 'S7O12', 'S7O13', 'S7O14', 'S7O15', 'S7O16',
'S7O17', 'S7O18'
    ]

    signal_levels = []
    for subprocess_name in subprocesses:
        file_name =
subprocess_name+'_'+str(bearing)+'_static_'+date_time
        subprocess_name = subprocess_name+"()"
        sleep(1)
        tempsignal_levels = []
        exec(subprocess_name)
        png_name = '/home/rasp/Tesis/screenshot.png'
        for _ in range (8):
            tempsignal_level = subprocess.check_output(command,
shell=True)
```

```
            tempsignal_level = tempsignal_level.decode().strip()
            tempsignal_level =
tempsignal_level.split(':')[1].strip().replace(' dBm', '')
            tempsignal_levels.append(int(tempsignal_level))


        average_signal_level = np.mean(tempsignal_levels)
        signal_levels.append(average_signal_level)



    if signal_levels:
        with open(output_file, 'w', newline='') as csvfile:
            fieldnames = ['File Name', 'signal level']
            writer = csv.DictWriter(csvfile,
fieldnames=fieldnames)
            writer.writeheader()
            writer.writerows([{'File Name': file_name[:5], 'signal
level': signal_level} for file_name, signal_level in
zip(subprocesses, signal_levels)])
        print(f"Signal disimpan di {output_file}")
    else:
        print("Tidak ada sinyal pada file input.")



if __name__ == '__main__':
    GPIO.cleanup()
    setup()
    main()
```

Program perhitungan DoA pada pengukuran di titik stasioner

```
import csv
import pandas as pd
import numpy as np
#from extract_values import extract_values
import math
import xml.etree.ElementTree as ET
import simplekml
import tkinter as tk
from tkinter import filedialog
import os
import re


def mean_square_error(data_actual, data_predicted):
    data_predicted = np.array(data_predicted, dtype=np.float64)
    diff = [actual - predicted for actual, predicted in
zip(data_actual, data_predicted)]
    squared_diff = [d ** 2 for d in diff]
    mse = sum(squared_diff) / len(squared_diff)
    return mse

# Menghapus data yang tidak memiliki koordinat dan tidak hasil
ukur yang tidak lengkap
def clean_data(data):
    df = pd.DataFrame(data)
    print(df.dtypes)
    # Menghapus baris dengan nilai n/a pada koordinat
    df.replace("n/a", np.nan, inplace=True)
    df = df.dropna(how='any')
    df = df.reset_index(drop=True)

    # Menemukan indeks awal pattern yang berurutan 1-8
    pattern_start_index =
df.index[df['JsonData.Pattern'].eq(1)].tolist()

    # Menyimpan indeks pattern yang berurutan 1-8
    pattern_indexes = []

    # Mengiterasi indeks awal pattern
    for start_index in pattern_start_index:
        pattern_range = [start_index]
        pattern_value = 2

        # Memeriksa angka berikutnya dalam pattern berurutan 1-8
        while pattern_value <= 8:
            next_index = pattern_range[-1] + 1

            # Memeriksa apakah angka berikutnya dalam pattern
berurutan
            if next_index >= len(df) or df.loc[next_index,
'JsonData.Pattern'] != pattern_value:
                break

            pattern_range.append(next_index)
```

```
            pattern_value += 1

        # Memeriksa apakah pattern berurutan 1-8 selesai
        if pattern_value == 9:
            pattern_indexes.extend(pattern_range)

    # Memfilter dataframe berdasarkan indeks pattern yang
berurutan 1-8
    df_filtered = df.loc[pattern_indexes]

    # Menampilkan hasil
    return df_filtered




def DoA(Calibrate, Measure):
    Numerator = Calibrate.dot(Measure)
    Denum1 = np.square(Calibrate).sum(axis=1)
    Denum1 = np.sqrt(Denum1)
    Denum2 = math.sqrt(Measure.dot(Measure))
    Denum = Denum1*Denum2
    CrossCorrelation = Numerator/(Denum)
    print("Nilai Cross Correlation Maksimum :",
np.max(CrossCorrelation))
    AngleDoa = np.argmax(CrossCorrelation)


    return AngleDoa, Numerator, Denum1, Denum2


def hitung_deviasi (target, realisasi):
    deviasi = np.mod((realisasi - target + 180), 360) - 180
    deviasi = np.where(deviasi >= 180, deviasi - 360, deviasi)
    return deviasi

def read_csv():
    # Membuka file dialog untuk memilih file CSV
    csv_path = filedialog.askopenfilename(filetypes=[("CSV Files",
"*.csv")])

    # Membaca file CSV menggunakan pandas
    dfRadiation = pd.read_csv(csv_path)



if __name__ == '__main__':
    Freq = 5560
    mac = "34:60:f9:6f:fe:6a"

    # Buat dialog pemilihan file
    root = tk.Tk()
    root.withdraw()
```

```
    # Membaca file hasil pengukuran degan pemilihan single freq
dan single mac
    MeasSingleFilePath =
filedialog.askopenfilename(filetypes=[("CSV", "*.csv")])

    # Membaca arah sumber pancaran berdasarkan nama file

    DoATarget = MeasSingleFilePath.split('_')[0]
    DoATarget = DoATarget.split("/")
    DoATarget = DoATarget[-1]
    DoATarget = DoATarget.strip()
    DoATarget = np.array(DoATarget, dtype=np.float64)
    print("DoA Target : ", DoATarget)
    pattern = []


    with open(MeasSingleFilePath, 'r') as meas:
        dataMeas = csv.DictReader(meas)
        #jumlah_data = len(list(dataMeas))-1

        Nama_konfigurasi = []
        signal_level = []

        for row in dataMeas:
            nama = row['File Name']  # Mengambil nilai kolom "File
Name"
            level = row['signal level']  # Mengambil nilai kolom
"signal level"

            Nama_konfigurasi.append(nama)
            signal_level.append(level)

    jumlah_data = len(signal_level)
    jumlah_loop = jumlah_data/8

    print(Nama_konfigurasi)
    print(signal_level)

    signal_level =np.array(signal_level,dtype='float')

    # Membaca file pola radiasi hasil kalibrasi
    # Mendapatkan direktori dari MeasSingleFilePath
    directory = os.path.dirname(MeasSingleFilePath)

    # Mendapatkan nama file dasar tanpa ekstensi
    filename = os.path.basename(MeasSingleFilePath)
    filename_without_extension = os.path.splitext(filename)[0]

    Denum2 = []
    NilaiDoa = []
    Num = []
    Denum1 = []
    Denum2 = []
    maxDBM = []
    MaxBeam = []
    ArrayTarget=[]
```

```
    # Melakukan loop untuk memnghitung DoA dan membandingkan
dengan tabel kalibrasi
    n = 0
    while n < jumlah_loop:
        tempRSS = signal_level[(0+(n*8)):(8+(n*8)):1]
        tempMaxRSSI = np.max(tempRSS)
        normalizedRSS = tempRSS-tempMaxRSSI
        print ("Temp RSS : ",tempRSS)
        TempMaxBeam = np.argmax(tempRSS)

        # Membuka file kalibrasi
        prefix = Nama_konfigurasi[(n*8)+1][:4]
        nama_file_kalibrasi = f"{prefix}_{Freq}.csv"
        print("Nama File Kalibrasi : ", nama_file_kalibrasi)
        dfRadiation = pd.read_csv(nama_file_kalibrasi)
        Pattern = dfRadiation[["1","2","3","4","5","6","7","8"]]
        Pattern = Pattern.reset_index(drop=True)
        print("Pattern :", Pattern)

        NormalizedPattern = Pattern.apply(lambda x: x - x.max(),
axis=0)
        NormalizedPattern =
NormalizedPattern.reset_index(drop=True)
        print("Norm Pattern :", NormalizedPattern)

        TempDoa, TempNum, TempDenum1,TempDenum2 =
DoA(NormalizedPattern,normalizedRSS) # output 10 data
        NilaiDoa.append(TempDoa)
        Num.append(TempNum)
        Denum1.append(TempDenum1)
        Denum2.append(TempDenum2)
        maxDBM.append(tempMaxRSSI)
        MaxBeam.append(determine_value(TempMaxBeam))
        ArrayTarget.append(DoATarget)

        n+=1


    print("DoA antena : ", NilaiDoa)
    print("Data yang dieksport untuk ditampilkan di peta: ")
    print("MAC : ", mac)
    print("SSID : ")
    print("Freq : ", Freq)

    print("Jumlah Doa = :",len(NilaiDoa))

    print("Max Beam = ", MaxBeam)
    print("Doa Target = ", DoATarget)

    # deviasi

    Deviasi =
hitung_deviasi(np.array(ArrayTarget),np.array(NilaiDoa))
    print ("Deviasi = ", Deviasi)
```

```
    # Mean square error untuk metode PPCC
    MSE_PPCC = mean_square_error(NilaiDoa,ArrayTarget)
    print ("Mean Square Error metode PPCC = ", MSE_PPCC)

    # Mean square error untuk metode Max Power Beam
    MSE_Power = mean_square_error(MaxBeam,ArrayTarget)
    print ("Mean Square Error metode maksimum beam = ", MSE_Power)

    # Menyimpan properti hasil pengukuran
    output_file = 'data_norm_' +
os.path.splitext(os.path.basename(MeasSingleFilePath))[0] + '.txt'
    output_file = os.path.join(directory, output_file)



    with open(output_file, 'w') as file:
        file.write(f'Bearing Antena : {DoATarget}\n')
        file.write(f'Konfigurasi Antena : {prefix}\n')
        file.write(f'Frekuensi : {Freq}\n')
        file.write(f'DoA algoritma PPCC  : {NilaiDoa}\n')
        file.write(f'Mean Square Error metode PPCC :
{MSE_PPCC}\n')
        file.write(f'DoA algoritma Beam Maksimum  : {MaxBeam}\n')
        file.write(f'Mean Square Error metode maksimum beam:
{MSE_Power}\n')
        file.write(f'Deviasi: {Deviasi}\n')

    # rekapitulasi hasil pengukuran
    rekapitulasi_file = 'Rekapitulasi.csv'
    rekapitulasi_file = os.path.join(directory, rekapitulasi_file)
    print("Rekapitulasi file : ", rekapitulasi_file)
    # Menyusun data yang akan ditambahkan sebagai baris baru
    data = np.hstack((DoATarget, NilaiDoa, Deviasi,
MeasSingleFilePath))
    print("DAta :", data)
    with open(rekapitulasi_file, 'a', newline='') as file:
        writer = csv.writer(file)
        writer.writerow(data)

    print ("Tersimpan")
    print(output_file)
```

Program pengukuran perangkat telekomunikasi RLAN secara bergerak

```
from gps3.agps3threaded import AGPS3mechanism
from i2c_hmc5883l import HMC5883
import gps
import os
import time
import sys
import json
import subprocess
import errno

import RPi.GPIO as GPIO
import time
import datetime
import smbus
import math
start_time = time.time()



SW1 = 11
SW2 = 13
SW3 = 15
SW4 = 19
SW5 = 21
SW6 = 23
SW7 = 29
SW8 = 31

def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(SW1,GPIO.OUT)
    GPIO.setup(SW2,GPIO.OUT)
    GPIO.setup(SW3,GPIO.OUT)
    GPIO.setup(SW4,GPIO.OUT)
    GPIO.setup(SW5,GPIO.OUT)
    GPIO.setup(SW6,GPIO.OUT)
    GPIO.setup(SW7,GPIO.OUT)
    GPIO.setup(SW8,GPIO.OUT)


def getWifi():
    command2 = 'sudo ip link set wlan0 up'
    command ='sudo iw dev wlan0 scan | egrep
"^BSS|signal|SSID|freq:"'
    command5 = 'sudo iw dev wlan0 scan freq 5560 flush | egrep
"^BSS|signal|SSID|freq:"'
    #c ommand5 = "timeout 5s bash -c \"sudo iw dev wlan0 scan freq
5600 flush | egrep '^BSS|signal|SSID|freq:'\""


    while(True):
        try:
            subprocess.check_output(command2, shell=True)
            result = subprocess.check_output(command5, shell=True)
```

```
            break
        except:
            continue

    result = str(result)[2::].replace('(on wlan0)',"")
    result = result.replace('\\n',' ').replace('\\b',' ')
    result = result.replace('\\t','').replace('dBm',
'').replace('SSID:','').replace('BSS','').replace('signal:','').
replace('freq:','').replace(' -- associated',"")

    result = result.split()[:-1]
    tempDict = {}
    wifiList = []
    prevKey = ''
    i = 0

    # MAC Address is key. Value is [dBm,SSId]
    while i <len(result)-2:

        if (result[i].count(':') == 5):
            #if tempDict:
            #    wifiList.append(tempDict)
            tempDict = {}
            tempDict['MAC'] = result[i]
            tempDict['Freq'] = result[i+1]
            tempDict['DBM'] = result[i+2]
            tempDict['SSID'] = result[i+3]
            i += 4
            if tempDict:
                wifiList.append(tempDict)
        # Apabila terdapat spasi pada SSID
        else:
            tempDict['SSID'] += ' ' + result[i+4]
            i += 1

    return wifiList


# Program untuk defenisi fungsi bearing kompas
rev = GPIO.RPI_REVISION
if rev == 2 or rev == 3:
    bus = smbus.SMBus(1)
else:
    bus = smbus.SMBus(0)

address = 0x1e

def read_byte(adr):
    return bus.read_byte_data(address, adr)

def read_word(adr):
    high = bus.read_byte_data(address, adr)
    low = bus.read_byte_data(address, adr+1)
    val = (high << 8) + low
    return val
```

```
def read_word_2c(adr):
    val = read_word(adr)
    if (val >= 0x8000):
        return -((65535 - val) + 1)
    else:
        return val

def write_byte(adr, value):
    bus.write_byte_data(address, adr, value)

def ant_configuration():
    argument = input("Masukkan konfigurasi antena : ")
    return argument

def S2O61():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 1')


def S2O62():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 2')


def S2O63():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S2O6..Arah 3')


def S2O64():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
```

```
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S2O6..Arah 4')


def S2O65():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 5')


def S2O66():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 6')


def S2O67():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 7')


def S2O68():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S2O6..Arah 8')
```

```
def S3O51():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 1')


def S3O52():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 2')


def S3O53():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 3')


def S3O54():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 4')


def S3O55():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
```

```
    GPIO.output(SW8,GPIO.HIGH)
    print('S3O5..Arah 5')


def S3O56():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 6')


def S3O57():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 7')


def S3O58():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S3O5..Arah 8')



def S4O41():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 1')


def S4O42():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
```

```
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 2')


def S4O43():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 3')


def S4O44():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 4')


def S4O45():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S4O4..Arah 5')


def S4O46():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 6')
```

```
def S4O47():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 7')


def S4O48():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S4O4..Arah 8')


def S5O31():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 1')


def S5O32():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 2')


def S5O33():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
```

```
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 3')


def S5O34():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 4')


def S5O35():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 5')


def S5O36():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S5O3..Arah 6')


def S5O37():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 7')


def S5O38():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
```

```
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S5O3..Arah 8')

def S6O21():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 1')

def S6O22():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 2')

def S6O23():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 3')

def S6O24():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 4')

def S6O25():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
```

```
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 5')

def S6O26():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S6O2..Arah 6')

def S6O27():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)
    print('S6O2..Arah 7')

def S6O28():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S6O2..Arah 8')

def S7O11():
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 1')

def S7O12():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
```

```
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 2')

def S7O13():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 3')

def S7O14():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 4')


def S7O15():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 5')

def S7O16():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 6')

def S7O17():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
```

```
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)
    print('S7O1..Arah 7')

def S7O18():
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)
    print('S7O1..Arah 8')

def S1O71():
    #print ('SW1 ON..')
    GPIO.output(SW1,GPIO.HIGH)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)


def S1O72():
    #print ('SW2 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.HIGH)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O73():
    #print ('SW3 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.HIGH)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O74():
    #print ('SW4 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
```

```
    GPIO.output(SW4,GPIO.HIGH)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O75():
    #print ('SW5 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.HIGH)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O76():
    #print ('SW6 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.HIGH)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.LOW)

def S1O77():
    #print ('SW7 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.HIGH)
    GPIO.output(SW8,GPIO.LOW)

def S1O78():
    #print ('SW8 ON..')
    GPIO.output(SW1,GPIO.LOW)
    GPIO.output(SW2,GPIO.LOW)
    GPIO.output(SW3,GPIO.LOW)
    GPIO.output(SW4,GPIO.LOW)
    GPIO.output(SW5,GPIO.LOW)
    GPIO.output(SW6,GPIO.LOW)
    GPIO.output(SW7,GPIO.LOW)
    GPIO.output(SW8,GPIO.HIGH)



if __name__ == '__main__':
    setup()
    koreksi = 4
    now = datetime.datetime.now()
```

```
    date_time = now.strftime("%m%d%H%M")
    element_configuration = ant_configuration()
    command2 = 'sudo gpsd /dev/ttyAMA0 -F /var/run/gpsd.sock'
    subprocess.check_output(command2, shell=True)
    command3 = 'sudo chmod 666 /dev/ttyAMA0'
    subprocess.check_output(command3, shell=True)

    agps_thread = AGPS3mechanism()# Instantiate AGPS3 Mechanisms
    agps_thread.stream_data() # From localhost (), or other hosts,
by example, (host='gps.ddns.net')
    agps_thread.run_thread() # Throttle time to sleep after an
empty lookup, default '()' 0.2 two tenths of a second

    time.sleep(3)


    outerDict = {"JsonData":[]}

    i = 0
    while 1:


        j = 1
        while j<9:
            y = element_configuration+str(j)+"()"
            exec (y)


            temp = agps_thread.data_stream

            data = {}
            data["wifi"] = getWifi()

            # Heading
            write_byte(0, 0b01110000) # Set to 8 samples @ 15Hz
            write_byte(1, 0b00100000) # 1.3 gain LSb / Gauss 1090
(default)
            write_byte(2, 0b00000000) # Continuous sampling

            scale = 0.92
            x_offset = -6.5
            y_offset = -521.5

            x_out = (read_word_2c(3) - x_offset) * scale
            y_out = (read_word_2c(7) - y_offset) * scale
            z_out = (read_word_2c(5)) * scale



            bearing  = math.atan2(y_out, x_out)
            if (bearing < 0):
                bearing += 2 * math.pi

            declination = 0.53
            bearing = math.degrees(bearing)
```

```
            bearing = math.fmod(360+(-1*bearing + koreksi), 360)
#tambahan 268
            bearing = bearing + declination

            data['Heading'] = bearing


            data['Latitude'] = temp.lat
            data['Longitude'] = temp.lon

            data['Pattern'] = j
            print("Iteration : d s s", (i, data['Latitude'],
data['Longitude'], data['Heading'], "Pattern : ",
data['Pattern']))
            j +=1

            i += 1
            outerDict["JsonData"].append(data)

            if (i == 1000):
                print("Menyimpan data..")

                with
open(date_time+'_track_'+element_configuration+'.json','w') as
outfile:
                    json.dump(outerDict, outfile, indent=4)
                print("Data telah berhasil disimpan")
                GPIO.cleanup()
                print("--- %s seconds ---" % (time.time() -
start_time))
                exit(1)
                i = 0
                outerDict["JsonData"] = []
```

Program untuk menghasilkan file kml untuk pemetaan DoA

```
from statistics import mean
import pandas as pd
import json
import numpy as np
#from extract_values import extract_values
import math
import xml.etree.ElementTree as ET
from shapely.geometry import Point, LineString
from geopy.point import Point as GeopyPoint
import simplekml
from geopy.distance import geodesic
import tkinter as tk
from tkinter import filedialog
import os
from polycircles import polycircles
import matplotlib.pyplot as plt
plt.rcParams["font.family"] = "Times New Roman"


def cari_pusat_lingkaran_terbanyak(lines, radius):
    max_intersections = 0
    selected_point = None

    # Menggunakan pendekatan skimming (sweep line) untuk
mengurutkan titik berdasarkan garis yang terdekat
    # print("Menggunakan pendekatan skimming (sweep line) untuk
mengurutkan titik berdasarkan garis yang terdekat")
    lines.sort(key=lambda line: line.distance(Point(0, 0)))

    for i, point in enumerate(lines):
        if i > 0 and lines[i-1].distance(point) > radius:
            # Menghentikan pencarian jika sudah melewati batas
radius
            break

        intersection_count = hitung_intersections(lines, point,
radius)
        if intersection_count > max_intersections:
            max_intersections = intersection_count
            selected_point = point


    return selected_point



def hitung_jarak_meter(lat1, lon1, lat2, lon2):
    point1 = (lat1, lon1)
    point2 = (lat2, lon2)
    distance_meter = geodesic(point1, point2).meters

    # return 0
    return distance_meter
```

```
def hitung_intersections(lines, point, radius):
    intersection_count = 0
    for line in lines:
        lon1, lat1 = line.coords[0]  # Ambil koordinat awal garis

        distance_to_start = hitung_jarak_meter(lat1, lon1,
point.y, point.x)
        if distance_to_start <= radius: #or distance_to_end <=
radius:
            intersection_count += 1


    return intersection_count

def hitung_deviasi (target, realisasi):
    deviasi = np.mod((realisasi - target + 180), 360) - 180
    deviasi = np.where(deviasi >= 180, deviasi - 360, deviasi)
    return deviasi

# Membuat objek KML
kml = simplekml.Kml()
kml2 = simplekml.Kml()
arrow_icon = kml.addfile('D:/00 ZASLI/01 OFFICE/S2/TE/03 Thesis/07
Raspberry/Tesis/arrow.png')

def mean_square_error(data_actual, data_predicted):
    diff = [actual - predicted for actual, predicted in
zip(data_actual, data_predicted)]
    squared_diff = [d ** 2 for d in diff]
    mse = sum(squared_diff) / len(squared_diff)
    return mse

# Menghapus data yang tidak memiliki koordinat dan tidak hasil
ukur yang tidak lengkap
def clean_data(data):
    df = pd.DataFrame(data)
    print(df.dtypes)
    # Menghapus baris dengan nilai n/a pada koordinat
    df.replace("n/a", np.nan, inplace=True)
    df = df.dropna(how='any',subset=['JsonData.Latitude',
'JsonData.Longitude'])
    #df2 = df1.dropna(subset=['JsonData.Longitude'])
    #df.dropna(how='any',inplace=True)
    df = df.reset_index(drop=True)

    # Menemukan indeks awal pattern yang berurutan 1-8
    pattern_start_index =
df.index[df['JsonData.Pattern'].eq(1)].tolist()

    # Menyimpan indeks pattern yang berurutan 1-8
    pattern_indexes = []

    # Mengiterasi indeks awal pattern
    for start_index in pattern_start_index:
        pattern_range = [start_index]
        pattern_value = 2
```

```
        # Memeriksa angka berikutnya dalam pattern berurutan 1-8
        while pattern_value <= 8:
            next_index = pattern_range[-1] + 1

            # Memeriksa apakah angka berikutnya dalam pattern
berurutan
            if next_index >= len(df) or df.loc[next_index,
'JsonData.Pattern'] != pattern_value:
                break

            pattern_range.append(next_index)
            pattern_value += 1

        # Memeriksa apakah pattern berurutan 1-8 selesai
        if pattern_value == 9:
            pattern_indexes.extend(pattern_range)

    # Memfilter dataframe berdasarkan indeks pattern yang
berurutan 1-8
    df_filtered = df.loc[pattern_indexes]

    # Menampilkan hasil
    return df_filtered


def groupedAvg(myData, N=8):
    averages = []
    for i in range(0, len(myData), N):
        chunk = myData[i:i+N]
        numeric_chunk = [x for x in chunk if isinstance(x, (int,
float))]
        average = sum(numeric_chunk) / len(numeric_chunk)
        averages.append(average)

    return averages

def hitung_rata_rata(data, jumlah_baris_per_kelompok):
    kelompok = []
    rata_rata = []

    for i, nilai in enumerate(data):
        kelompok.append(nilai)

        if len(kelompok) == jumlah_baris_per_kelompok:
            rata_rata.append(sum(kelompok) / len(kelompok))
            kelompok = []

    if kelompok:
        rata_rata.append(sum(kelompok) / len(kelompok))

    return rata_rata


def DoA(Calibrate, Measure):
    Numerator = Calibrate.dot(Measure)
```

```
    #print("Numerator : ",Numerator)
    Denum1 = np.square(Calibrate).sum(axis=1)
    #print("Denum1a :",Denum1)
    Denum1 = np.sqrt(Denum1)
    #print("Denum1b :",Denum1)
    Denum2 = math.sqrt(Measure.dot(Measure))
    #print("Denumerator 2 :", Denum2)
    Denum = Denum1*Denum2
    #print("Denumereator = ", Denum)
    CrossCorrelation = Numerator/(Denum)
    #print("Nilai Cross Correlation Maksimum :",
np.max(CrossCorrelation))
    AngleDoa = np.argmax(CrossCorrelation)


    return AngleDoa, Numerator, Denum1, Denum2,CrossCorrelation

def determine_value(number):
    value_dict = {
        0: 0,
        1: 45,
        2: 88,
        3: 135,
        4: 180,
        5: 225,
        6: 270,
        7: 315
    }
    if number in value_dict:
        return value_dict[number]
    else:
        return "Nilai tidak valid"

# Fungsi untuk mengonversi sudut dalam derajat menjadi radian
def to_radians(degrees):
    return degrees * (math.pi / 180)

# Fungsi untuk menghitung sudut antara dua titik koordinat
def calculate_bearing(lat1, lon1, lat2, lon2):
    # Konversi koordinat ke dalam radian
    lat1_rad = to_radians(lat1)
    lon1_rad = to_radians(lon1)
    lat2_rad = to_radians(lat2)
    lon2_rad = to_radians(lon2)

    # Perhitungan selisih longitude dan latitude
    delta_lon = lon2_rad - lon1_rad

    # Perhitungan sudut menggunakan formula haversine
    y = math.sin(delta_lon) * math.cos(lat2_rad)
    x = math.cos(lat1_rad) * math.sin(lat2_rad) -
math.sin(lat1_rad) * math.cos(lat2_rad) * math.cos(delta_lon)
    bearing_rad = math.atan2(y, x)

    # Konversi sudut ke dalam derajat
    bearing_deg = (math.degrees(bearing_rad) + 360) % 360
```

```
    return bearing_deg

def read_csv():
    # Membuka file dialog untuk memilih file CSV
    csv_path = filedialog.askopenfilename(filetypes=[("CSV Files",
"*.csv")])

    # Membaca file CSV menggunakan pandas
    dfRadiation = pd.read_csv(csv_path)



if __name__ == '__main__':
    Freq = 5560
    mac = "34:60:f9:6f:fe:6a"

    # Buat dialog pemilihan file
    root = tk.Tk()
    root.withdraw()


    # Membaca file hasil pengukuran degan pemilihan single freq
dan single mac
    # Memilih file JSON
    MeasSingleFilePath =
filedialog.askopenfilename(filetypes=[("JSON Files", "*.json")])


    with open(MeasSingleFilePath, 'r') as meas:
        dataMeas = json.load(meas)

    jsonwifi = pd.json_normalize(dataMeas,
record_path=['JsonData','wifi'],
        meta=[
            ['JsonData','Heading'],
            ['JsonData','Latitude'],
            ['JsonData','Longitude'],
            ['JsonData','Pattern'],
        ])
    print(jsonwifi)
    jsonwifi = jsonwifi[(jsonwifi["MAC"]==mac)]
    print("Filtered mac =", jsonwifi)
    print("Panda JsonWifi :", pd.DataFrame(jsonwifi))

    # Tambahan untuk cleansing
    # Menghapus baris dengan nilai n/a

    jsonwifi= clean_data(jsonwifi)
    #jsonwifi= clean_data(jsonwifi)
    print("Cleaned jsonwifi :", jsonwifi)

    # menghitung rata-rata koordinat untuk masing-masing
konfigurasi antena
    latitude = jsonwifi['JsonData.Latitude']
    longitude = jsonwifi['JsonData.Longitude']
```

```
    heading = jsonwifi['JsonData.Heading']


    meanLong = hitung_rata_rata(longitude,8)
    meanLat = hitung_rata_rata(latitude,8)
    meanHead = hitung_rata_rata(heading,8)


    RSS = jsonwifi["DBM"]
    RSS = RSS.reset_index(drop=True)
    print("RSS Filtered= ", RSS)

    # Mengambil RSS setiap 8 data koordinat
    # https://numpy.org/doc/stable/user/basics.indexing.html
    RSS = np.array(RSS, dtype='float')
    print ("RSS1 =", RSS )
    RSS1= RSS[0:8:1]

    # Membaca file pola radiasi hasil kalibrasi
    # Mendapatkan direktori dari MeasSingleFilePath
    directory = os.path.dirname(MeasSingleFilePath)

    # Mendapatkan nama file dasar tanpa ekstensi
    filename = os.path.basename(MeasSingleFilePath)
    filename_without_extension = os.path.splitext(filename)[0]

    # Mendapatkan 4 karakter terakhir dari nama file sebagai
config_ant
    config_ant = filename_without_extension[-4:]

    # Menggabungkan direktori dengan filename
    filename_with_directory = os.path.join(directory, filename)

    # Membuka dan memproses file dfRadiation
    filename_csv = f"{config_ant}_{Freq}.csv"
    dfRadiation = pd.read_csv(os.path.join(directory,
filename_csv))
    Pattern = dfRadiation[["1","2","3","4","5","6","7","8"]]
    Pattern = Pattern.reset_index(drop=True)

    print(Pattern)
    NormalizedPattern = Pattern.apply(lambda x: x - x.max(),
axis=0)
    NormalizedPattern = NormalizedPattern.reset_index(drop=True)
    print("Norm Pattern :", NormalizedPattern)



    # Melakukan perhitungan DoA dengan algoritma Power Pattern
Cross Correlation
    #print("Jumlah loop = ", len(RSS)/8)
    CountLoop = len(RSS)/8
    Denum2 = []
    Doa = []
    Num = []
    Denum1 = []
```

```
    Denum2 = []
    CrossCorr = []
    CrossCorr2 = []
    maxDBM = []
    MaxBeam = []
    sudut = list(range(361))

    n = 0

    while n<CountLoop:
        tempRSS = RSS[(0+(n*8)):(8+(n*8)):1]
        tempMaxRSSI = np.max(tempRSS)
        tempMinRSSI = np.min(tempRSS)
        normalizedRSS = tempRSS-tempMaxRSSI
        if tempMaxRSSI == tempMinRSSI:
            normalizedRSS = np.ones(8)
        TempMaxBeam = np.argmax(tempRSS)
        TempDoa, TempNum, TempDenum1,TempDenum2,CrossCorr  =
DoA(NormalizedPattern,normalizedRSS) # output 10 data

        #Menyimpan crossccorrelation ke dalam file csv
        hasil2 = {'Sudut': sudut, 'Cross Correlation' : CrossCorr}
        hasil_cross = pd.DataFrame(hasil2)

        csv_hasil_cc = 'CC_'+ str(n+1) +'_' +
os.path.splitext(os.path.basename(MeasSingleFilePath))[0] + '.csv'
        csv_hasil_cc = os.path.join(directory, csv_hasil_cc)
        hasil_cross.to_csv(csv_hasil_cc, index=False)

        # Membuat plot cross correlation
        plt.figure(figsize=(8, 5))
        #plt.plot(hasil_cross['Sudut'], hasil_cross['Cross
Correlation'], marker='o')
        plt.plot(hasil_cross['Sudut'], hasil_cross['Cross
Correlation'], label='Cross Correlation')

        plt.xlabel('Sudut')
        plt.ylabel('Cross Correlation')
        #plt.title('Plot Cross Correlation vs Sudut')

        # Menandai nilai maksimum pada plot
        print(" data ke : ", n)
        print("Cross correlation : ", CrossCorr)
        max_corr_index = hasil_cross['Cross Correlation'].idxmax()
        max_corr_value = hasil_cross.loc[max_corr_index, 'Cross
Correlation']
        max_corr_angle = max_corr_index


        print("max cross index = ", max_corr_index)
        print("max cross value = ", max_corr_value)
        print("max cross sudut = ", max_corr_angle)

        # Format angka sudut tanpa angka di belakang koma
        formatted_angle = '{:.0f}'.format(max_corr_angle)
```

```
        # Menambahkan marker pada titik maksimum
        plt.scatter(max_corr_angle, max_corr_value, color='red',
marker='o', label=f'Max: ({max_corr_angle:.2f},
{max_corr_value:.2f})')


        # Menambahkan label pada titik maksimum
        plt.text(max_corr_angle, max_corr_value, f'Max:
({formatted_angle}, {max_corr_value:.2f})', fontsize=12,
                 verticalalignment='bottom',
horizontalalignment='left')



        # Menyimpan file gambar
        cross_plot_filename = 'plot'+ str(n+1) +'_' +
os.path.splitext(os.path.basename(MeasSingleFilePath))[0] + '.jpg'
        cross_plot_filename = os.path.join(directory,
cross_plot_filename)
        plt.savefig(cross_plot_filename)
        plt.close()


        Doa.append(TempDoa)
        Num.append(TempNum)
        Denum1.append(TempDenum1)
        Denum2.append(TempDenum2)
        maxDBM.append(tempMaxRSSI)
        MaxBeam.append(determine_value(TempMaxBeam))

        n+=1

    Correction=0

    print("Doa : ",Doa)

    NorthDoA = []
    for head, doa in zip(meanHead, Doa):
        north_doa = (head + doa) % 360
        NorthDoA.append(north_doa)

    print("heading : ", meanHead)
    print("DoA antena : ", Doa)
    print("North Doa = ", NorthDoA)
    print("Data yang dieksport untuk ditampilkan di peta: ")
    print("MAC : ", mac)
    print("SSID : ")
    print("Freq : ", Freq)
    print("Lintang : ", meanLat)
    print("Bujur : ", meanLong)
    print("RSSI : ", maxDBM)
    print("Jumlah max dbm = :",len(maxDBM))
    print("Jumlah Doa = :",len(Doa))
    print("MeanHead = :",len(meanHead))
    print("Max Beam = ", MaxBeam)
```

```
    latitude_target = 0.577031
    longitude_target = 123.048227
    DoA_target = []
    deviasi_PPCC = []
    lines = []
    jarak_pengukuran = []

    # print("Loop melalui setiap data dan tambahkan markah dengan
simbol panah pada KML")
    # Loop melalui setiap data dan tambahkan markah dengan simbol
panah pada KML
    for i in range(len(meanLat)):
        # Create placemark
        #placemark = kml.newpoint(name=f'DoA_PPCC {i+1}',
description=f'Heading: {meanHead[i]}\nRSSI: {maxDBM[i]}\nDoA:
{NorthDoA[i]}')
        placemark = kml.newpoint(name=f'', description=f'Heading:
{meanHead[i]}\nRSSI: {maxDBM[i]}\nDoA: {NorthDoA[i]}')
        placemark.coords = [(meanLong[i], meanLat[i], 0)]
        placemark.style.iconstyle.href = arrow_icon
        placemark.style.iconstyle.heading = meanHead[i]

        # Persiapan mebuat garis
        # Ambil koordinat titik awal
        latitude_start = meanLat[i]
        longitude_start = meanLong[i]

        # Buat objek Point menggunakan shapely
        #start_point = Point(longitude_start, latitude_start)
        start_point = Point(latitude_start, longitude_start)

        # Hitung koordinat akhir (latitude, longitude) menggunakan
geopy
        azimuth_degrees = NorthDoA[i] #Doa[i]  # Arah dalam
derajat
        distance_meters = 400  # Jarak dalam meter
        azimuth_radians = math.radians(azimuth_degrees)  # Ubah ke
radian

        # Hitung koordinat akhir menggunakan geopy
        delta_latitude =
distance_meters*math.cos(azimuth_radians)/111320.0 #111111
        delta_longitude =
distance_meters*math.sin(azimuth_radians)/(111320.0*math.cos(math.
radians(meanLat[i])))


        # Ambil koordinat akhir
        latitude_end = latitude_start+delta_latitude
        longitude_end = longitude_start+delta_longitude

        # Mendapatkan koordinat lintang dan bujur baru
        #latitude_end = destination_point.latitude
        #longitude_end = destination_point.longitude
```

```
        # Tentukan warna berdasarkan rentang nilai maxDBM
        if maxDBM[i] < -90:
            color = simplekml.Color.rgb(128, 0, 128, 64) # purple
#'800080'  # Ungu
        elif maxDBM[i] >= -90 and maxDBM[i] < -80:
            color = simplekml.Color.rgb(0, 0, 255, 96) #blue
#'0000ff'  # Biru
        elif maxDBM[i] >= -80 and maxDBM[i] < -70:
            color = simplekml.Color.rgb(255, 255, 0, 128) #yellow
#'ffff00'  # Kuning
        elif maxDBM[i] >= -70 and maxDBM[i] < -60:
            color = simplekml.Color.rgb(255, 165, 0, 160) #orange
#'ffa500'  # Orange
        else:
            color = simplekml.Color.rgb(255, 0, 0, 192)#red
#'ff0000'  # Merah


        linestring = kml.newlinestring(name=f'Line {i+1}',
coords=[(meanLong[i], meanLat[i], 0), (longitude_end,
latitude_end, 0)])
        lines.append([(meanLong[i], meanLat[i]), (longitude_end,
latitude_end)])

        linestring.style.linestyle.color=color
        linestring.style.linestyle.width=2

        #mengitung sudut yang seharusnya

        # Hitung sudut antara titik A dan B
        temp_DoA_target = calculate_bearing(latitude_start,
longitude_start, latitude_target, longitude_target)
        tempdeviasi_PPCC =
hitung_deviasi(temp_DoA_target,azimuth_degrees)
        deviasi_PPCC.append(tempdeviasi_PPCC)
        DoA_target.append(temp_DoA_target)

        # menghitung jarak pengukuran terhadap pemancar
        temp_jarak_pengukuran =
hitung_jarak_meter(meanLat[i],meanLong[i],latitude_target,longitud
e_target)
        jarak_pengukuran.append(temp_jarak_pengukuran)

    # print("Algoritma bisection untuk memperoleh perpotongan
garis")
    # Algoritma bisection untuk memperoleh perpotongan garis
    radius_lingkaran = 10
    intersection_points = []
    coord_intersection_point = []
    for i in range(len(lines)):
        line_coords = lines[i]

        # Buat objek LineString untuk garis saat ini
        line = LineString(line_coords)
```

```
        # Loop melalui garis-garis lainnya untuk mencari
perpotongan
        for j in range(i + 1, len(lines)):
            other_line_coords = lines[j]
            other_line = LineString(other_line_coords)

            # Cek apakah garis saat ini dan garis lainnya memiliki
titik perpotongan
            if line.intersects(other_line):
                intersection = line.intersection(other_line)
                intersection_points.append(intersection)

                # Tambahkan titik perpotongan sebagai markah pada
KML
                placemark = kml.newpoint(coords=[(intersection.x,
intersection.y, 0)])
                placemark.style.iconstyle.icon.href =
simplekml.IconStyle().icon.href
                placemark.style.iconstyle.hotspot.x = 0.5  #
Koordinat X hotspot (0.5 adalah tengah)
                placemark.style.iconstyle.hotspot.y = 0.5  #
Koordinat Y hotspot (0.5 adalah tengah)
                placemark.style.iconstyle.scale = 0.5
                placemark.style.iconstyle.color = 'ff0000ff'  #
Titik berwarna biru

                # Hitung sudut antara titik perpotongan dan target
                latitude_start, longitude_start =
intersection.coords[0]
                coord_intersection_point.append((intersection.x,
intersection.y))
                #azimuth_degrees =
calculate_bearing(latitude_start, longitude_start,
latitude_target, longitude_target)



    #-----------------
    print("koordinat perpotongan : ", coord_intersection_point)
    print("DAta dari intersection point : ",intersection_points)

    selected_point =
cari_pusat_lingkaran_terbanyak(intersection_points,10)

    # Buat lingkaran di sekitar titik terpilih
    longitude_cent, latitude_cent = selected_point.coords[0]
    kml.newpoint(name=f'Hasil Perhitungan Lokasi Sumber Pancaran',
coords=[(longitude_cent, latitude_cent, 0)])

    # Atur properti style untuk lingkaran
    #polygon_area.style.polystyle.color =
simplekml.Color.changealphaint(100, simplekml.Color.rgb(255, 0,
0))
```

```
    polycircle =
polycircles.Polycircle(latitude=latitude_cent,longitude=longitude_
cent, radius=radius_lingkaran,number_of_vertices=60)
    circle = kml.newpolygon(name="Area perkiraan sumber pancaran",
outerboundaryis=polycircle.to_kml())
    circle.style.polystyle.color =
simplekml.Color.changealphaint(200,simplekml.Color.rgb(0,0,255,64)
)


    output_kml1 = os.path.join(directory,
f"3intersect_{filename_without_extension}_DoA_PPCC.kml")#"intersec
tion_{filename_without_extension}_DoA_PPCC.kml"
#os.path.join(directory,
f"intersct_{filename_without_extension}_DoA_PPCC.kml")
    kml.save(output_kml1)

    print ("Intersection point : ", intersection_points)
    print("Data has been exported to a KML file.")

    print("Tesrimpan.", filename_without_extension)
    print("DoA relatif PPCC : ", Doa)
    print("DoA arah utara PPCC: ", NorthDoA)
    print("Doa Target :", DoA_target)
    print("deviasi PPCC: ", deviasi_PPCC)
    rmse_PPCC = np.sqrt(np.mean(np.square(deviasi_PPCC)))
    print("RMSE PPCC: ", rmse_PPCC)

    # menghitung selisih jarak perhitungan dengan jarak real
    Selisih_jarak =
hitung_jarak_meter(latitude_cent,longitude_cent,latitude_target,lo
ngitude_target)
    print("hasil perhitungan lokasi sumber pancaran pada koordinat
: ", latitude_cent,", ",longitude_cent )
    print("Selisih jarak dari lokasi pemancar : ", Selisih_jarak)


    # Menyimpan properti hasil pengukuran
    stat_file = '3data_norm_' +
os.path.splitext(os.path.basename(MeasSingleFilePath))[0] + '.txt'
    stat_file = os.path.join(directory, stat_file)


    with open(stat_file, 'w') as file:
        file.write(f'Konfigurasi Antena : {config_ant}\n')
        file.write(f'Frekuensi : {Freq}\n')
        file.write(f'DoA target  : {DoA_target}\n')
        file.write(f'DoA algoritma PPCC  : {Doa}\n')
        file.write(f'DoA terhadap utara  : {NorthDoA}\n')
        file.write(f'Deviasi  : {deviasi_PPCC}\n')
        file.write(f'Mean Square Error metode PPCC :
{rmse_PPCC}\n')
        file.write(f'Koordinat Hasil Perhitungan  :
{latitude_cent}, {longitude_cent}\n')
        file.write(f'Deviasi: {Selisih_jarak} meter\n')
```

```
    #Menyimpan hasil ke dalam file csv
    hasil = {'Lintang' : meanLat,'Bujur' : meanLong,'Heading' :
meanHead,'Jarak Pengukuran' : jarak_pengukuran,'DoA Target':
DoA_target,'DoA Relatif' : Doa,'DoA
Pengukuran':NorthDoA,'Deviasi':deviasi_PPCC}
    hasil_olah = pd.DataFrame(hasil)
    csv_hasil_file = 'hasil_' +
os.path.splitext(os.path.basename(MeasSingleFilePath))[0] + '.csv'
    csv_hasil_file = os.path.join(directory, csv_hasil_file)
    hasil_olah.to_csv(csv_hasil_file, index=False)
```