

## DAFTAR PUSTAKA

- Aliyu, I., Gana, K. J., Musa, A. A., Agajo, J., Orire, A. M., Abiodun, F. T., & Adegbeye, M. A. (2017). A PROPOSED FISH COUNTING ALGORITHM USING DIGITAL IMAGE PROCESSING. *Journal of Science, Technology & Education (JOSTE)*, V.
- Almarinez, J. H., & Hernandez, A. (2019). Evaluation of Mangrove Crab Classification System. *International Journal of Recent Technology and Engineering (IJRTE)*, 2277-3878.
- Fierro, A. N., Camarillo, D. R., & Miyatake, M. N. (2017). Mosquito larva classification method based on convolutional neural networks. *International Conference on Electronics, Communications and Computers (CONIELECOMP)*.
- French, G., Fisher, M. H., Mackiewicz, M., & Needle, C. (2015). Convolutional Neural Networks for Counting Fish in Fisheries Surveillance Video. *Workshop on Machine Vision of Animals and their Behaviour*.
- Kaewchote, J., Janyong, S., & Limprasert, W. (2018). Image recognition method using Local Binary Pattern and the Random forest classifier to count post larvae shrimp. *Agriculture and Natural Resources*.
- Munir, R. (2004). *Pengolahan Citra Digital Dengan Pendekatan Algoritmik*. Bandung: INFORMATIKA.
- Putra, J. G. (2019). *Pengenalan Konsep Pembelajaran Mesin dan Deep Learning*. Tokyo.
- Raman, V., Perumal, S., Navaratnam, S., & Fazilah, S. (2015). Computer Assisted Counter System for Larvae and Juvenile Fish in Malaysian Fishing Hatcheries by Machine Learning Approach. *Journal of Computers*, 423-431.
- Redmon, J., & Farhadi, A. (2018). YOLOV3: An Incremental Improvement. *arXiv*.
- Redmon, J., Vala, S. D., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *arXiv*.



## LAMPIRAN

### Lampiran 1. Source code training YOLO V3 dengan framework darknet

```
#!/python3
"""
Python 3 wrapper for identifying objects in images
Requires DLL compilation
Both the GPU and no-GPU version should be compiled; the no-GPU
version should be renamed "yolo_cpp_dll_nogpu.dll".
On a GPU system, you can force CPU evaluation by any of:
- Set global variable DARKNET_FORCE_CPU to True
- Set environment variable CUDA_VISIBLE_DEVICES to -1
- Set environment variable "FORCE_CPU" to "true"

@author: Philip Kahn
@date: 20180503
"""
#pylint: disable=R, W0401, W0614, W0703
from ctypes import *
import math
import random
import os

def sample(probs):
    s = sum(probs)
    probs = [a/s for a in probs]
    r = random.uniform(0, 1)
    for i in range(len(probs)):
        r = r - probs[i]
        if r <= 0:
            return i
    return len(probs)-1

def array(ctype, values):
```



```

arr = (ctype*len(values))()
arr[:] = values
return arr

class BOX(Structure):
    _fields_ = [("x", c_float),
                ("y", c_float),
                ("w", c_float),
                ("h", c_float)]

class DETECTION(Structure):
    _fields_ = [("bbox", BOX),
                ("classes", c_int),
                ("prob", POINTER(c_float)),
                ("mask", POINTER(c_float)),
                ("objectness", c_float),
                ("sort_class", c_int)]

class IMAGE(Structure):
    _fields_ = [("w", c_int),
                ("h", c_int),
                ("c", c_int),
                ("data", POINTER(c_float))]

class METADATA(Structure):
    _fields_ = [("classes", c_int),
                ("names", POINTER(c_char_p))]

#lib = CDLL("/home/pjreddie/documents/darknet/libdarknet.so",
            RTLD_GLOBAL)
lib = CDLL("libdarknet.so", RTLD_GLOBAL)
lib = True
name == "nt":

```



```

    cwd = os.path.dirname(__file__)
    os.environ['PATH'] = cwd + ';' + os.environ['PATH']
    winGPUdll = os.path.join(cwd, "yolo_cpp_dll.dll")
    winNoGPUdll = os.path.join(cwd, "yolo_cpp_dll_nogpu.dll")
    envKeys = list()
    for k, v in os.environ.items():
        envKeys.append(k)
    try:
        try:
            tmp = os.environ["FORCE_CPU"].lower()
            if tmp in ["1", "true", "yes", "on"]:
                raise ValueError("ForceCPU")
            else:
                print("Flag value '"+tmp+"' not forcing CPU mode")
        except KeyError:
            # We never set the flag
            if 'CUDA_VISIBLE_DEVICES' in envKeys:
                if int(os.environ['CUDA_VISIBLE_DEVICES']) < 0:
                    raise ValueError("ForceCPU")
            try:
                global DARKNET_FORCE_CPU
                if DARKNET_FORCE_CPU:
                    raise ValueError("ForceCPU")
            except NameError:
                pass
            # print(os.environ.keys())
            # print("FORCE_CPU flag undefined, proceeding with
GPU")
            if not os.path.exists(winGPUdll):
                raise ValueError("NoDLL")
            lib = CDLL(winGPUdll, RTLD_GLOBAL)
        except (KeyError, ValueError):
            hasGPU = False
            if os.path.exists(winNoGPUdll):

```



```

        lib = CDLL(winNoGPUdll, RTLD_GLOBAL)
        print("Notice: CPU-only mode")
    else:
        # Try the other way, in case no_gpu was
        # compile but not renamed
        lib = CDLL(winGPUdll, RTLD_GLOBAL)
        print("Environment variables indicated a CPU run, but
we didn't find '"+winNoGPUdll+"`. Trying a GPU run anyway.")
    else:
        lib = CDLL("./libdarknet.so", RTLD_GLOBAL)
lib.network_width.argtypes = [c_void_p]
lib.network_width.restype = c_int
lib.network_height.argtypes = [c_void_p]
lib.network_height.restype = c_int

copy_image_from_bytes = lib.copy_image_from_bytes
copy_image_from_bytes.argtypes = [IMAGE, c_char_p]

def network_width(net):
    return lib.network_width(net)

def network_height(net):
    return lib.network_height(net)

predict = lib.network_predict_ptr
predict.argtypes = [c_void_p, POINTER(c_float)]
predict.restype = POINTER(c_float)

if hasGPU:
    set_gpu = lib.cuda_set_device
    set_gpu.argtypes = [c_int]
    make_image = lib.make_image
    make_image.argtypes = [c_int, c_int, c_int]

```



```

make_image.restype = IMAGE

get_network_boxes = lib.get_network_boxes
get_network_boxes.argtypes = [c_void_p, c_int, c_int, c_float,
c_float, POINTER(c_int), c_int, POINTER(c_int), c_int]
get_network_boxes.restype = POINTER(DETECTION)

make_network_boxes = lib.make_network_boxes
make_network_boxes.argtypes = [c_void_p]
make_network_boxes.restype = POINTER(DETECTION)

free_detections = lib.free_detections
free_detections.argtypes = [POINTER(DETECTION), c_int]

free_ptrs = lib.free_ptrs
free_ptrs.argtypes = [POINTER(c_void_p), c_int]

network_predict = lib.network_predict_ptr
network_predict.argtypes = [c_void_p, POINTER(c_float)]

reset_rnn = lib.reset_rnn
reset_rnn.argtypes = [c_void_p]

load_net = lib.load_network
load_net.argtypes = [c_char_p, c_char_p, c_int]
load_net.restype = c_void_p

load_net_custom = lib.load_network_custom
load_net_custom.argtypes = [c_char_p, c_char_p, c_int, c_int]
load_net_custom.restype = c_void_p

```

```

obj = lib.do_nms_obj
obj.argtypes = [POINTER(DETECTION), c_int, c_int, c_float]

```



```

do_nms_sort = lib.do_nms_sort
do_nms_sort.argtypes = [POINTER(DETECTION), c_int, c_int, c_float]

free_image = lib.free_image
free_image.argtypes = [IMAGE]

letterbox_image = lib.letterbox_image
letterbox_image.argtypes = [IMAGE, c_int, c_int]
letterbox_image.restype = IMAGE

load_meta = lib.get_metadata
lib.get_metadata.argtypes = [c_char_p]
lib.get_metadata.restype = METADATA

load_image = lib.load_image_color
load_image.argtypes = [c_char_p, c_int, c_int]
load_image.restype = IMAGE

rgbgr_image = lib.rgbgr_image
rgbgr_image.argtypes = [IMAGE]

predict_image = lib.network_predict_image
predict_image.argtypes = [c_void_p, IMAGE]
predict_image.restype = POINTER(c_float)

def array_to_image(arr):
    import numpy as np
    # need to return old values to avoid python freeing memory
    arr = arr.transpose(2,0,1)
    c = arr.shape[0]
    arr.shape[1]
    arr.shape[2]
    = np.ascontiguousarray(arr.flat, dtype=np.float32) / 255.0

```



```

    data = arr.ctypes.data_as(POINTER(c_float))
    im = IMAGE(w,h,c,data)
    return im, arr

def classify(net, meta, im):
    out = predict_image(net, im)
    res = []
    for i in range(meta.classes):
        if altNames is None:
            nameTag = meta.names[i]
        else:
            nameTag = altNames[i]
        res.append((nameTag, out[i]))
    res = sorted(res, key=lambda x: -x[1])
    return res

def detect(net, meta, image, thresh=.5, hier_thresh=.5, nms=.45,
debug= False):
    """
    Performs the meat of the detection
    """
    #pylint: disable= C0321
    im = load_image(image, 0, 0)
    if debug: print("Loaded image")
    ret = detect_image(net, meta, im, thresh, hier_thresh, nms,
debug)
    free_image(im)
    if debug: print("freed image")
    return ret

def detect_image(net, meta, im, thresh=.5, hier_thresh=.5,
debug= False):
    import cv2
    custom_image_bgr = cv2.imread(image) # use:
    , imagePath,)

```





```

    #custom_image = cv2.cvtColor(custom_image_bgr,
cv2.COLOR_BGR2RGB)

    #custom_image =
cv2.resize(custom_image, (lib.network_width(net),
lib.network_height(net)), interpolation = cv2.INTER_LINEAR)

    #import scipy.misc

    #custom_image = scipy.misc.imread(image)

    #im, arr = array_to_image(custom_image)
# you should comment line below: free_image(im)

    num = c_int(0)
    if debug: print("Assigned num")
    pnum = pointer(num)
    if debug: print("Assigned pnum")
    predict_image(net, im)
    if debug: print("did prediction")

    #dets = get_network_boxes(net, custom_image_bgr.shape[1],
custom_image_bgr.shape[0], thresh, hier_thresh, None, 0, pnum, 0)
# OpenCV

    dets = get_network_boxes(net, im.w, im.h, thresh, hier_thresh,
None, 0, pnum, 0)

    if debug: print("Got dets")
    num = pnum[0]
    if debug: print("got zeroth index of pnum")
    if nms:
        do_nms_sort(dets, num, meta.classes, nms)
    if debug: print("did sort")
    res = []
    if debug: print("about to range")
    for j in range(num):
        if debug: print("Ranging on "+str(j)+" of "+str(num))
        if debug: print("Classes: "+str(meta), meta.classes,
meta.names)

    for i in range(meta.classes):
        if debug: print("Class-ranging on "+str(i)+" of
meta.classes)+"= "+str(dets[j].prob[i]))
        if dets[j].prob[i] > 0:
            b = dets[j].bbox

```



```

        if altNames is None:
            nameTag = meta.names[i]
        else:
            nameTag = altNames[i]
        if debug:
            print("Got bbox", b)
            print(nameTag)
            print(dets[j].prob[i])
            print((b.x, b.y, b.w, b.h))
        res.append((nameTag, dets[j].prob[i], (b.x, b.y,
b.w, b.h)))
        if debug: print("did range")
        res = sorted(res, key=lambda x: -x[1])
        if debug: print("did sort")
        free_detections(dets, num)
        if debug: print("freed detections")
        return res

netMain = None
metaMain = None
altNames = None

```

```

def performDetect(imagePath="data/dog.jpg", thresh= 0.25,
configPath = "./cfg/yolov3.cfg", weightPath = "yolov3.weights",
metaPath= "./cfg/coco.data", showImage= True, makeImageOnly =
False, initOnly= False):

```

```

    """

```

```

        Convenience function to handle the detection and returns of
objects.

```

```

        Displaying bounding boxes requires libraries scikit-image and
numpy

```



```

        parameters

```

```

        imagePath: str

```

```

    Path to the image to evaluate. Raises ValueError if not
found

    thresh: float (default= 0.25)
        The detection threshold

    configPath: str
        Path to the configuration file. Raises ValueError if not
found

    weightPath: str
        Path to the weights file. Raises ValueError if not found

    metaPath: str
        Path to the data file. Raises ValueError if not found

    showImage: bool (default= True)
        Compute (and show) bounding boxes. Changes return.

    makeImageOnly: bool (default= False)
        If showImage is True, this won't actually *show* the
image, but will create the array and return it.

    initOnly: bool (default= False)
        Only initialize globals. Don't actually run a prediction.

```

Returns

-----

When showImage is False, list of tuples like

```

('obj_label', confidence, (bounding_box_x_px,
bounding_box_y_px, bounding_box_width_px, bounding_box_height_px))

```

The X and Y coordinates are from the center of the bounding box. Subtract half the width or height to get the lower



```

Otherwise, a dict with
    {
        "detections": as above
        "image": a numpy array representing an image,
compatible with scikit-image
        "caption": an image caption
    }
"""
# Import the global variables. This lets us instance Darknet
once, then just call performDetect() again without instancing
again

global metaMain, netMain, altNames #pylint: disable=W0603

assert 0 < thresh < 1, "Threshold should be a float between
zero and one (non-inclusive)"

if not os.path.exists(configPath):
    raise ValueError("Invalid config path
`"+os.path.abspath(configPath)+"`)")

if not os.path.exists(weightPath):
    raise ValueError("Invalid weight path
`"+os.path.abspath(weightPath)+"`)")

if not os.path.exists(metaPath):
    raise ValueError("Invalid data file path
`"+os.path.abspath(metaPath)+"`)")

if netMain is None:
    netMain = load_net_custom(configPath.encode("ascii"),
weightPath.encode("ascii"), 0, 1) # batch size = 1

if metaMain is None:
    metaMain = load_meta(metaPath.encode("ascii"))

if altNames is None:
    # In Python 3, the metafile default access craps out on
Windows (but not Linux)

    # Read the names file and create a list to feed to detect
try:
    with open(metaPath) as metaFH:
        metaContents = metaFH.read()

    import re

```



```

        match = re.search("names *= *(.*)$", metaContents,
re.IGNORECASE | re.MULTILINE)
        if match:
            result = match.group(1)
        else:
            result = None
        try:
            if os.path.exists(result):
                with open(result) as namesFH:
                    namesList =
namesFH.read().strip().split("\n")
                    altNames = [x.strip() for x in
namesList]
            except TypeError:
                pass
        except Exception:
            pass
    if initOnly:
        print("Initialized detector")
        return None
    if not os.path.exists(imagePath):
        raise ValueError("Invalid image path
`"+os.path.abspath(imagePath)+"`)")
    # Do the detection
    #detections = detect(netMain, metaMain, imagePath, thresh)
    # if is used cv2.imread(image)
    detections = detect(netMain, metaMain,
imagePath.encode("ascii"), thresh)
    if showImage:
        try:
            from skimage import io, draw
            import numpy as np
            image = io.imread(imagePath)
            print("*** "+str(len(detections))+ " Results, color
confidence ***")
            imcaption = []

```



```

for detection in detections:
    label = detection[0]
    confidence = detection[1]
    pstring = label+": "+str(np rint(100 *
confidence))+"%"
    imcaption.append(pstring)
    print(pstring)
    bounds = detection[2]
    shape = image.shape
    # x = shape[1]
    # xExtent = int(x * bounds[2] / 100)
    # y = shape[0]
    # yExtent = int(y * bounds[3] / 100)
    yExtent = int(bounds[3])
    xExtent = int(bounds[2])
    # Coordinates are around the center
    xCoord = int(bounds[0] - bounds[2]/2)
    yCoord = int(bounds[1] - bounds[3]/2)
    boundingBox = [
        [xCoord, yCoord],
        [xCoord, yCoord + yExtent],
        [xCoord + xExtent, yCoord + yExtent],
        [xCoord + xExtent, yCoord]
    ]
    # Wiggle it around to make a 3px border
    rr, cc = draw.polygon_perimeter([x[1] for x in
boundingBox], [x[0] for x in boundingBox], shape= shape)
    rr2, cc2 = draw.polygon_perimeter([x[1] + 1 for x
in boundingBox], [x[0] for x in boundingBox], shape= shape)
    rr3, cc3 = draw.polygon_perimeter([x[1] - 1 for x
in boundingBox], [x[0] for x in boundingBox], shape= shape)
    rr4, cc4 = draw.polygon_perimeter([x[1] for x in
gBox], [x[0] + 1 for x in boundingBox], shape= shape)
    rr5, cc5 = draw.polygon_perimeter([x[1] for x in
gBox], [x[0] - 1 for x in boundingBox], shape= shape)

```



```

        boxColor = (int(255 * (1 - (confidence ** 2))),
int(255 * (confidence ** 2)), 0)
        draw.set_color(image, (rr, cc), boxColor, alpha= 0.8)
        draw.set_color(image, (rr2, cc2), boxColor, alpha= 0.8)
        draw.set_color(image, (rr3, cc3), boxColor, alpha= 0.8)
        draw.set_color(image, (rr4, cc4), boxColor, alpha= 0.8)
        draw.set_color(image, (rr5, cc5), boxColor, alpha= 0.8)
        if not makeImageOnly:
            io.imshow(image)
            io.show()
        detections = {
            "detections": detections,
            "image": image,
            "caption": "\n<br/>".join(imcaption)
        }
    except Exception as e:
        print("Unable to show image: "+str(e))
    return detections

if __name__ == "__main__":
    print(performDetect())

```



## Lampiran 2. Source code deteksi dan visualisasi larva dengan YOLO V3

```
# import packages

import numpy as np

import argparse

import time

import cv2

import os

# construct the argument parse and parse the arguments

ap = argparse.ArgumentParser()

ap.add_argument("-i", "--image", required=True,

                help="path to input image")

ap.add_argument("-c", "--confidence", type=float, default=0.2,

                help="minimum probability to filter weak detections")

ap.add_argument("-t", "--threshold", type=float, default=0.5,

                help="threshold when applying non-maxima suppression")

args = vars(ap.parse_args())

# load the YOLO class labels

labelsPath = "data/obj.names"

LABELS = open(labelsPath).read().strip().split("\n")

# paths to the YOLO weights and model configuration

weightsPath = "newweight/small-larva_2000.weights"

path = "cfg/small-larva.cfg"
```





```

# initialize a list of colors to represent each possible class
label

COLORS = (103, 220, 225)

# np.random.seed(42)

# COLORS = np.random.randint(0, 255, size=(len(LABELS), 3),
#     dtype="uint8")

# load our YOLO object detector
print("Processing...")

net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

# load input image and grab its spatial dimensions
image = cv2.imread(args["image"])

# image = cv2.resize(image, (0,0), fx=0.7, fy=0.7)

(H, W) = image.shape[:2]

# determine only the *output* layer names that we need from YOLO
ln = net.getLayerNames()

ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]

# construct a blob from the input image and then perform a forward
# pass of the YOLO object detector, giving us our bounding boxes
and
# associated probabilities

blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (416,
416), (128, 128, 128), swapRB=True, crop=False)

net.setInput(blob)

start = time.time()

```



```

layerOutputs = net.forward(ln)

end = time.time()

print("Nilai blob: {}".format(blob.shape))

# initialize our lists of detected bounding boxes, confidences,
and
# class IDs, respectively
boxes = []
confidences = []
classIDs = []

# loop over each of the layer outputs
for output in layerOutputs:
    # loop over each of the detections
    for detection in output:
        # extract the class ID and confidence (i.e.,
probability) form Image
        scores = detection[5:]
        classID = np.argmax(scores)
        confidence = scores[classID]
        # print ("Nilai Confidence dari object ialah :",
confidence)
        # print ("Nilai ID dari Class ialah :", classID)

# filter out weak predictions by ensuring the detected
# probability is greater than the minimum probability
if confidence > args["confidence"]:
```



```

# scale the bounding box coordinates back
relative to the size of the image

box = detection[0:4] * np.array([W, H, W, H])
(centerX, centerY, width, height) =
box.astype("int")

# print ("Nilai dari B.Box ialah :", centerX,"
", centerY, " ", width, " ", height)

# use the center (x, y)-coordinates to get the
top and and left

#corner of the bounding box
x = int(centerX - (width / 2))
y = int(centerY - (height / 2))
print ("Nilai x dan y dari B.Box ialah :", x,"
", y)

# update our list of bounding box coordinates,
confidences, and class IDs

boxes.append([x, y, int(width), int(height)])
confidences.append(float(confidence))
classIDs.append(classID)

# apply non-maxima suppression to suppress weak, overlapping
bounding boxes

idxs = cv2.dnn.NMSBoxes(boxes, confidences, args["confidence"],
["threshold"])

```

```

if len(idxs) > 0:
    # at least one detection exists

```



```

if len(idxs) > 0:
    # loop over the indexes
    for i in idxs.flatten():
        # extract the bounding box coordinates
        (x, y) = (boxes[i][0], boxes[i][1])
        (w, h) = (boxes[i][2], boxes[i][3])

        # draw a bounding box rectangle and label on the image
        # color = [int(c) for c in COLORS[classIDs[i]]]
        cv2.rectangle(image, (x, y), (x + w, y + h), COLORS,
8)

        text = "{}: {:.4f}".format(LABELS[classIDs[i]],
confidences[i])

        print(text)

        cv2.putText(image, text, (x, y - 5),
cv2.FONT_HERSHEY_SIMPLEX, 2, COLORS, 8)

        # cv2.putText(image, text, (x, y - 5),
cv2.FONT_HERSHEY_SIMPLEX, 0.2, COLORS, 2)

# Font type
font = cv2.FONT_HERSHEY_SIMPLEX

# Font Coordinate
org = (40, 480)

# Font Size
fontScale = 10

# Font color with format (B,G,R)
(206, 0, 0)
Thickness

```



```
thickness = 20

image = cv2.putText(image, '{} larva '.format(len(idxs)), org,
font, fontScale, color, thickness, cv2.LINE_AA)

# Write total of larva in the frame into larvaNumber.txt
with open("jumlahlarva.txt", "a") as myfile:
    myfile.write("Ada {} Larva yang
terdeteksi\n".format(len(idxs)))

# show timing information on YOLO
print("YOLO took {:.6f} seconds".format(end - start))

# Save the output image
# cv2.imshow("Image.jpg", image)
cv2.imwrite("Image.jpg", image)
print ("ada", len(idxs), "Larva")
cv2.waitKey(0)
```



### Lampiran 3. Source code counting larva dengan YOLO V3

```
import os

from os import walk

import PIL

from PIL import Image

import subprocess

import sys

import datetime

import time

os.system

start = time.time()

img_dir = "images/"

out_dir = "output/"

text_file = open('imgpath.txt', 'w')

#List of Images Path

txt_img_list = []

#Iteration for showing pictures that we want to process

for (dirpath, dirnames, filenames) in walk(img_dir):

    txt_img_list.extend(filenames)

    break

print(txt_img_list)

text_file.write("\n")

#Process

for txt_name in txt_img_list:

    #Save images path

    text_file.write('images/%s.jpg\n'%(os.path.splitext(txt_name) [0]))
```



```

text_file.close()

#Variable Init
d = 1
total = 0

#Reading image path
with open('imgpath.txt', 'r') as fobj:
    for line in fobj:
        image_List = [[num for num in line.split()] for line in
fobj]

open("larvaNumber.txt", "w").close()

#Iteration for detecting images
for images in image_List:
    commands = ["python", "MyYOLO2.py", "-i", images[0]]
    # os.system(' '.join(commands))
    os.system(' '.join(commands))
    output = subprocess.check_output(commands)
    output = output.decode("utf-8").split("\n")

    #Count the number of lines that contain "larva
    numLarva = len([i.split(":")[0] for i in output if
i.split(":")[0] == 'larva'])

    # print("Ada {} larva yang terdeteksi.".format(numLarva))

#Write total of larva in the frame into larvaNumber.txt
with open("larvaNumber.txt", "a") as myfile:
    myfile.write("Pada gambar ke {} Ada {} Larva yang
ksi\n".format(d, numLarva))

```



```
#Save predictions image into output folder
predicted_image = Image.open("Image.jpg")
output_image = "output/predicted_image%3d.jpg"%d
predicted_image.save(output_image)

d+=1

end = time.time()

# print("WAKTU PROSES adalah ",end - start, "detik")
total=total+numLarva

end = time.time()
print("TOTAL WAKTU PROSES adalah ",end - start, "detik")
print("TOTAL ADA ", total, "LARVA")
```







## SURAT PERSETUJUAN

Nomor : 32845/UN4.1.1.2.1.1/PK.02.03/2019


Berdasarkan Peraturan Rektor Universitas Hasanuddin tentang Penyelenggaraan Program Sarjana Nomor : 2781/UN4.1/KEP/2018 TANGGAL 16 Juli 2018, dengan ini menerangkan bahwa :

Nama : MUH. ARIEF WICAKSONO  
Tempat/Tanggal Lahir : UJUNG PANDANG / 22 JANUARI 1997  
Stambuk : D42115302  
Fakultas : TEKNIK  
Program Studi : TEKNIK INFORMATIKA

Telah memenuhi syarat untuk Ujian Skripsi Strata I (S1). Demikian Surat Persetujuan ini dibuat untuk digunakan dalam proses pelaksanaan ujian skripsi, dengan ketentuan dapat mengikuti Wisuda PERIODE III MARET 2020. Jika persyaratan kelulusan/wisuda telah dipenuhi. Terima Kasih.

Makassar, 19 Desember 2019

a.n. Kepala Bagian Akademik  
Universitas Hasanuddin,

  
Mursalim, S.Sos  
Nip. 19730216199601 1 001



### Keterangan :

Nomor Use : D42115302  
Nomor Password/Pin : 32030833  
Alamat Websit : <http://unhas.ac.id/akad/wisuda/>  
Layanan E-Mail : [alimkomath@gmail.com](mailto:alimkomath@gmail.com)  
Catatan :

1. Bagi Mahasiswa yang telah melaksanakan Ujian Sarjana dan dinyatakan lulus, segera menyerahkan lembar pengesahan Skripsi dan Berita Acara Ujian Sarjana ke Sub Bagian Akademik Fakultas, untuk memperoleh nomor Alumni dan didaftar sebagai Wisudawan pada periode berjalan.
2. Jika terjadi perubahan Judul Skripsi agar melaporkan ke kasubag. Pendidikan Fakultas sebelum didaftar sebagai Wisudawan pada Periode berjalan
3. Pada saat ON - LINE Mahasiswa diharapkan mengisi Identitas diri sesuai Surat Izin Ujian ini.
4. Surat izin ujian hanya berlaku untuk Wisuda periode berjalan (Wisuda Maret 2020)





KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI  
UNIVERSITAS HASANUDDIN

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK INFORMATIKA

Kampus Fakultas Teknik Unhas, Jl. Poros Malino, Gowa  
<http://eng.unhas.ac.id/informatika>, Email : [informatika@unhas.ac.id](mailto:informatika@unhas.ac.id)

DAFTAR HADIR SEMINAR HASIL

Nama/Stambuk : 1.Muh.Arief Wicaksono D421 15302  
Judul Skripsi/T.A : "Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection"  
Hari/Tanggal : Kamis, 12 Desember 2019  
Jam : 10-00 Wita – Selesai  
Tempat : Ruang Lab.UBICON Departemen Teknik Informatika Gowa

No.	Jabatan	Nama Dosen	Tanda Tangan
L.	Pembimbing I	1. Dr.Eng.Muhammad Niswar,ST.,M.I.T	1.....
	Pembimbing II	2. Dr. Eng. Intan Sari Areni,ST.,M.T	2.....
II.	Anggota Penguji	3. Dr.Amil Ahmad Ilham,ST.,M.I.T	3.....
		4. Anugrayani Bustamin,ST.,M.T	4.....

PANITIA UJIAN

Ketua,

Dr.Eng.Muhammad Niswar,ST.,M.I.T

Sekretaris,

Dr. Eng. Intan Sari Areni,ST.,M.T





KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI  
UNIVERSITAS HASANUDDIN

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK INFORMATIKA

Kampus Fakultas Teknik Unhas, Jl. Poros Malino, Gowa  
<http://eng.unhas.ac.id/informatika>, Email : [informatika@unhas.ac.id](mailto:informatika@unhas.ac.id)

**BERITA ACARA SEMINAR HASIL**

Pada hari ini **Kamis**, tanggal **12 Desember, 2019** Pukul **10.00 WITA** - **Selesai** bertempat di **Ruang Lab.UBICON Departemen Teknik Informatika Gowa**, telah dilaksanakan Seminar Hasil bagi Saudara :

Nama : Muh.Arief Wicaksono  
No. Stambuk : D421 15302  
Fakultas/Departemen : Teknik/Teknik Informatika  
Judul Skripsi : "Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection"

Yang dihadiri oleh Tim Penguji Seminar Hasil sebagai berikut :

No.	Nama	Jabatan	Tanda tangan
1.	Dr.Eng.Muhammad Niswar,ST.,M.I.T	Pemb I/Ketua	1...
2.	Dr. Eng. Intan Sari Areni,ST.,M.T	Pemb II/Sekretaris	2...
3.	Dr.Amil Ahmad Ilham,ST.,M.I.T	Anggota	3...
4.	Anugrayani Bustamin,ST.,M.T	Anggota	4...

Hasil keputusan Tim Penguji Seminar Hasil : **Lulus / Tidak lulus** dengan nilai angka .....**87**.....  
dan huruf .....**A**.....

Makassar, 12 Desember 2019  
Ketua/Sekretaris Panitia Ujian,

Dr.Eng.Muhammad Niswar,ST.,M.I.T





KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI  
UNIVERSITAS HASANUDDIN

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK INFORMATIKA

Kampus Fakultas Teknik Unhas, Jl. Poros Malino, Gowa  
<http://eng.unhas.ac.id/informatika>, Email : [informatika@unhas.ac.id](mailto:informatika@unhas.ac.id)

**SURAT KETERANGAN NILAI SEMINAR HASIL**

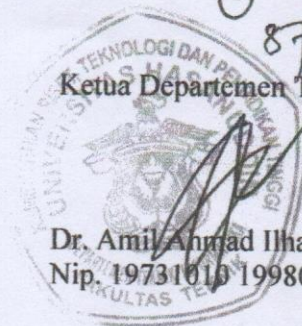
Nomor : 730 / UN4.7.7.TI/PK.03.06/2019

Pada hari ini Kamis, tanggal 12 Desember, 2019 Pukul 10.00 WITA - Selesai bertempat di Ruang Lab. UBICON Departemen Teknik Informatika, telah dilaksanakan Seminar Hasil bagi Saudara :

Nama : Muh.Arief Wicaksono  
No. Stambuk : D421 15302  
Fakultas/Departemen : Teknik/Teknik Informatika  
Judul Skripsi : "Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection"

Setelah pembawa seminar hasil menguraikan tugas akhirnya dan menjawab pertanyaan dari Tim Penguji dinyatakan Lulus / Tidak Lulus dengan nilai :

6 A — B — B — C — C — D — E



Ketua Departemen Tek.Informatika,

Dr. Amil Anmad Ilham, ST., M.I.T  
Nip. 19731010 199802 1 001

Dosen Penguji,

Dr. Eng. Muhammad Niswar, ST., M.I.T  
Nip. 19730922 199903 1 001



Diketahui oleh,  
a.n Dekan,  
Wakil Dekan Bidang Akademik, Riset dan Inovasi

Prof. Baharuddin Hamzah, ST., M.Arch., Ph.D  
Nip. 19690308 199512 1 001



Optimization Software:  
[www.balesio.com](http://www.balesio.com)



KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI  
UNIVERSITAS HASANUDDIN  
FAKULTAS TEKNIK

Poros Malino Km.6Bontomarannu(92172) Gowa, Sulawesi Selatan 92172, Sulawesi Selatan  
Telp. (0411) 586015, 586262 Fax (0411) 586015  
<http://eng.unhas.ac.id>, Email : [teknik@unhas.ac.id](mailto:teknik@unhas.ac.id)

**SURAT PENUGASAN**  
No. 21571/UN4.7.1/TD.06/2019

Dari : Dekan Fakultas Teknik Universitas Hasanuddin.

Kepada : Mereka yang tercantum namanya di bawah ini.

Isi : 1. Bahwa berdasarkan peraturan Akademik Universitas Hasanuddin Tahun 2003 Pasal 36 butir 3 point a, b (SK. Rektor Unhas Nomor : 1067 /J04/PP.08/2008), dengan ini menugaskan Saudara sebagai PANITIA UJIAN SARJANA Program Strata Satu (S1) Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin dengan susunan sebagai berikut :

Pembimbing I / Ketua : 1. Dr.Eng. Muhammad Niswar, ST., M.IT  
Pembimbing II / Sekretaris : 2. Dr.Eng. Intan Sari Areni, ST., M.T  
Anggota : 3. Dr. Amil Ahmad Ilham, ST., M.IT  
4. Anugrayani Bustamin, ST., M.T

untuk menguji bagi mahasiswa tersebut di bawah ini :

Nama/NIM : Muhammad Arief Wicaksono D421 15 302  
Program Studi : Teknik Informatika  
Judul Thesis/Skripsi : " **Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection** "

2. Waktu Ujian ditetapkan oleh Panitia Ujian Sarjana Program Strata Satu (S1).
3. Agar Surat penugasan ini dilaksanakan sebaik-baiknya dengan penuh rasa tanggung jawab.
4. Surat penugasan ini berlaku sejak tanggal ditetapkan sampai dengan berakhirnya Ujian Sarjana tersebut, dengan ketentuan bahwa segala sesuatunya akan ditinjau dan diperbaiki sebagaimana mestinya apabila dikemudian hari ternyata terdapat kekeliruan dalam keputusan ini.

Ditetapkan di Gowa,  
Pada tanggal 23 Desember 2019  
a.n. Dekan

Wakil Dekan Bidang Akademik

Prof. Baharuddin Hamzah, ST., M.Arch., Ph.D  
NIP.1969030819951210011



Teknik Unhas  
Departemen Teknik Informatika FT-UH  
Manajemen dan Perlengkapan FT-UH






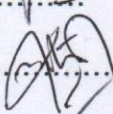
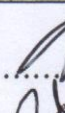
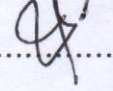
KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI  
UNIVERSITAS HASANUDDIN  
FAKULTAS TEKNIK

DEPARTEMEN TEKNIK INFORMATIKA

Kampus Fakultas Teknik Unhas, Jl. Poros Malino, Gowa  
<http://eng.unhas.ac.id/informatika>, Email : [informatika@unhas.ac.id](mailto:informatika@unhas.ac.id)

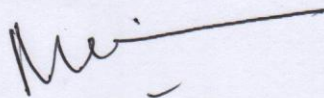
DAFTAR HADIR UJIAN SKRIPSI MAHASISWA  
FAKULTAS TEKNIK UNHAS

Nama/Stambuk : Muh.Arief Wicaksono D421 15302  
Judul Skripsi/T.A : "Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection"  
Hari/Tanggal : Senin, 30 Desember 2019  
Jam : 13.00 Wita – Selesai  
Tempat : Lab.UBICON Departemen Teknik Informatika Gowa

No.	Jabatan	Nama Dosen	Tanda Tangan
L.	Pembimbing I	1. Dr.Eng.Muhammad Niswar,ST.,M.I.T	1..... 
	Pembimbing II	2. Dr. Eng. Intan Sari Areni,ST.,M.T	2..... 
II.	Anggota Penguji	3. Dr.Amil Ahmad Ilham,ST.,M.I.T	3..... 
		4. Anugrayani Bustamin,ST.,M.T	4..... 

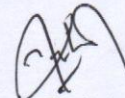
PANITIA UJIAN

Ketua,



Dr.Eng.Muhammad Niswar,ST.,M.I.T

Sekretaris,



Dr. Eng. Intan Sari Areni,ST.,M.T





**BERITA ACARA UJIAN SKRIPSI**

Pada hari ini **Senin**, tanggal **30 Desember, 2019** Pukul **13.00 WITA** - **Selesai** bertempat di **Lab.UBICON Departemen Teknik Informatika Gowa** , telah dilaksanakan Ujian Skripsi bagi Saudara :

Nama : Muh.Arief Wicaksono  
No. Stambuk : D421 15302  
Program Studi : Teknik Informatika  
Judul Skripsi : **“Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection“**

Yang dihadiri oleh Tim Penguji Ujian Skripsi sebagai berikut :

No.	Nama	Jabatan	Tanda tangan
1.	Dr.Eng.Muhammad Niswar,ST.,M.I.T	Pemb I/Ketua	1...
2.	Dr. Eng. Intan Sari Areni,ST.,M.T	Pemb II/Sekretaris	2...
3.	Dr. Amil Ahmad Ilham,ST.,M.I.T	Anggota	3...
4.	Anugrayani Bustamin,ST.,M.T	Anggota	4...

Hasil keputusan Tim Penguji Ujian Skripsi/Tugas Akhir : **Lulus** / ~~Tidak lulus~~ dengan nilai angka .......... dan huruf ..........

Gowa, 30 Desember, 2019

Ketua/Sekretaris Panitia Ujian,

Dr.Eng.Muhammad Niswar,ST.,M.I.T





KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI  
UNIVERSITAS HASANUDDIN  
FAKULTAS TEKNIK

DEPARTEMEN TEKNIK INFORMATIKA

Kampus Fakultas Teknik Unhas, Jl. Poros Malino, Gowa  
<http://eng.unhas.ac.id/informatika>, Email : [informatika@unhas.ac.id](mailto:informatika@unhas.ac.id)

SURAT KETERANGAN NILAI UJIAN SKRIPSI

Nomor : 033 / UN4.7.7.TI/PK.03.06/2019

Pada hari ini **Senin**, tanggal **30 Desember, 2019** Pukul **13.00 WITA** - **Selesai** bertempat di **Lab.UBICON Departemen Teknik Informatika Gowa**, telah dilaksanakan Ujian Skripsi bagi Saudara :

Nama : Muh.Arief Wicaksono  
No. Stambuk : D421 15302  
Program Studi : Teknik Informatika  
Judul Skripsi : **“Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode Object Detection“**

Setelah pembawa ujian Skripsi menguraikan tugas akhirnya dan menjawab pertanyaan dari Tim Penguji dinyatakan Lulus / Tidak Lulus dengan nilai :

(A) — A — B+ — B — B- — C+ — C — D — E —  
87

Mengetahui:

Ketua Departemen Tek.Informatika,

Dr. Amil Ahmad Ilham, ST., M.I.T  
Nip. 19731010 199802 1 001

Dosen Penguji,

Dr.Eng.Muhammad Niswar, ST., M.I.T  
Nip. 19730922 199903 1 001

Diketahui oleh,  
a.n. Dekan,  
Wakil Dekan Bidang Akademik, Riset dan Inovasi

Prof. Baharuddin Hamzah, ST., M.Arch., Ph.D  
Nip. 19690308 199512 1 001

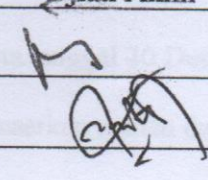















## KARTU BIMBINGAN SKRIPSI

Prodi S1 Teknik Informatika Universitas Hasanuddin

Stb.	Nama Mahasiswa
D42115302	Muh. Arief Wicaksono

Pembimbing.	Nama Pembimbing	Paraf & Tgl. Persetujuan Ujian Akhir
I	Dr.Eng. Muhammad Niswar,S.T., M.I.T	
II	Dr.Eng. Intan Sari Areni, S.T., M.T.	
No SK Pemb:		

Judul Skripsi:	Deteksi dan Perhitungan Jumlah Larva Kepiting Rajungan dengan Metode <i>Object Detection</i>
----------------	----------------------------------------------------------------------------------------------

No.	Tanggal Bimbingan	Uraian Kegiatan Bimbingan	Paraf Pemb.
1	27/06/2019	Konsultasi Pengambilan Data di SPBA P	
2	15/07/2019	Konsultasi Algoritma yg digunakan	
3	26/07/2019	Konsultasi Data gambar yang digunakan	
4	12/08/2019	Asistensi hasil deteksi Sistem.	
5	23/09/2019	Konsultasi Sistem Counting	
6	18/11/2019	Asistensi Counting larva.	
7	02/12/2019	Asistensi Flowchart Sistem.	
8	04/12/2019	Asistensi Parameter training	
9	09/12/2019	Presentasi persiapan Seminar hasil.	
10		Asistensi terkait Akurasi Sistem	
11		Presentasi hasil Revisi.	



# LEMBAR PERBAIKAN SKRIPSI

## “DETEKSI DAN PERHITUNGAN JUMLAH LARVA KEPITING RAJUNGAN DENGAN METODE OBJECT DETECTION”

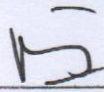

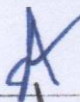

OLEH:

**MUH. ARIEF WICAKSONO**  
**NIM D42115302**

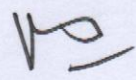
Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 30 Desember 2019.

Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr. Eng. Muhammad Niswar, ST., M.IT.	
Sekretaris	Dr.Eng. Intan Sari Areni, ST., M.T.	
Anggota	Dr. Amil Ahmad Ilham, S.T., M.IT.	
	Anugrayani Bustamin, S.T., M.T.	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Eng. Muhammad Niswar, ST., M.IT.	
II	Dr.Eng. Intan Sari Areni, ST., M.T.	