p-ISSN : 2443-2210
e-ISSN : 2443-2229

Jurnal Teknik Informatika dan Sistem Informasi
Volume 6 Nomor 3 Desember 2020

Deteksi Buah Menggunakan *Supervised Learning* dan Ekstraksi Fitur untuk Pemeriksa Harga

http://dx.doi.org/10.28932/jutisi.v6i3.3029

Kristiawan[#1], Deon Diamanta[#2], Try Atmaja[#3], Andreas Widjaja✉[#4]

[#]*Jurusan Magister Ilmu Komputer,Universitas Kristen Maranatha*
Jl. Surya Sumantri no. 65, Sukawarna

[1]mi1979005@student.it.maranatha.edu
[2]mi1979004@student.it.maranatha.edu
[3]mi1979003@student.it.maranatha.edu
[4]andreas.widjaja@it.maranatha.edu

Abstract — **The role of technology like Computer Vision in the business world is growing over time. This is no exception to the supermarket business. This study aims to build a prototype to help supermarket customers perform independent service and provide efficiency for companies engaged in the supermarket sector. This system can detect and recognize fruits using a standard camera and provide price information to users. This system utilizes the k-Nearest Neighbors and Histogram of Oriented Gradient algorithms to recognize fruit objects. By using these two algorithms, feature extraction can be done in the form of color and shape of the object which is then classified. This study takes the fruit images as input and then the recognition system shows the fruit name and price information.**

Keywords— **fruit detection; HoG; k-NN; OpenCV**

I. PENDAHULUAN

Peran serta teknologi dalam dunia bisnis kian besar, seiring dengan perkembangannya. Hal ini dimaksudkan untuk mempermudah proses-proses yang sering dilakukan dalam bisnis itu sendiri. Kini peran teknologi dapat kita jumpai dalam berbagai macam model bisnis, tidak berpaku pada dunia TI saja. Bisnis transportasi, bisnis makanan, hingga bisnis pengiriman barang dibuat jadi lebih mudah dan sederhana dengan adanya peran teknologi dalam model bisnisnya.

Bisnis lainnya seperti *supermarket* juga tidak luput dari adanya integrasi teknologi di dalamnya. Hal yang paling sering kita lihat seperti penggunaan mesin kasir yang pada waktu dulu hanya sekedar untuk menghitung transaksi konsumen dan menyimpan uang pembayaran dalam bentuk uang tunai, sekarang dengan adanya penerapan teknologi, mesin kasir terhubung dengan mesin *Electronic Data Capture* (*EDC*), *barcode scanner*, hingga *Near Field Communication* untuk pembayaran [1].

Peran teknologi di dalam bisnis, selain mempermudah para pelaku bisnis dalam menjalankan kegiatan operasional bisnisnya, juga memiliki tujuan untuk meningkatkan efisiensi. Teknologi mendorong dan memungkinkan pemrosesan data yang lebih cepat, pengambilan informasi yang lebih mudah, dan dalam beberapa kasus – otomatisasi yang diberikan oleh teknologi dapat mengurangi atau bahkan menggantikan peran karyawan / tenaga manusia secara fisik.

Ketika manusia melakukan tugas, itu bisa memakan waktu dan memiliki potensi untuk terjadinya kesalahan manusia (*human error*). Ketika teknologi digunakan untuk proses kerja / operasional yang berulang, peluang untuk terjadinya kesalahan dapat dikurangi atau bahkan dihilangkan. Demikian juga dengan waktu yang dibutuhkan untuk menyelesaikan tugas dapat dipersingkat. Selain membuat proses jadi berjalan lebih cepat, teknologi juga mempermudah untuk memperoleh informasi terkini.

Melihat keberagaman teknologi yang semakin bertambah dan terintegrasi kepada hal yang tadinya sederhana namun terbilang rumit dan memakan waktu untuk dilakukan, tidak menutup kemungkinan hal seperti itu dapat diterapkan kepada hal-hal lainnya seperti alat *price checker* yang juga sering ditemui pada *supermarket*. Dengan teknologi berupa *computer vision* untuk mengenali objek, peran manusia dapat dikurangi atau dihilangkan [2].

Seiring waktu, tenaga manusia juga akan bertambah mahal. Sebagai opsi dalam pengembangan bisnis, tentunya lebih baik mencari opsi pengganti yang menawarkan biaya lebih rendah untuk jangka panjang, dengan kata lain menawarkan efisiensi. Dalam konteks ini, peran teknologi dalam bisnis juga menjadi pilihan yang menguntungkan.

Teknologi *computer vision* pada alat *price checker* di

supermarket seperti yang disebutkan di atas memungkinkan konsumen yang berbelanja dapat melakukan kegiatan *self service* dengan melakukan *scan* barang kepada alat *computer vision* yang disediakan yang langsung dapat mengenali produk dan menunjukkan harga barang tersebut. Hal ini dapat dilakukan tanpa adanya peran serta dari karyawan seperti pada umumnya [3].

Konsumen tidak perlu mengetahui kode dari objek yang ingin dihitung. Konsumen hanya menunjukkan buah ke hadapan kamera, kamera akan mengenali objek tersebut dan memunculkan keterangan beserta harga dari buah tersebut. Dalam penelitian ini, objek akan difokuskan kepada buah-buahan. Penelitian ini mengarah kepada hal tersebut, walaupun masih beberapa langkah ke depan, yang akan dibuat baru pengenal buah saja [4].

Hasil dari penelitian ini adalah sebuah *prototype* sistem kamera yang menerima *input* berupa foto buah-buahan. Kamera ini terhubung dengan komputer / *laptop*, jadi setiap ada *input* berupa foto buah, *input* tersebut akan dimasukkan ke dalam algoritma *machine learning*. Kemudian pada hasil proses yang dikeluarkan oleh *machine learning* tersebut akan dilakukan *mapping* dengan *database* pada sistem. Kemudian harga dari buah tersebut akan muncul.

A. Rumusan Masalah

Bagaimana membuat sebuah sistem yang mampu mengenali buah-buahan dan membantu penghitungan harga dengan mudah bagi konsumen?

B. Batasan Masalah

Pembatasan masalah digunakan untuk menghindari adanya penyimpangan maupun pelebaran pokok masalah agar penelitian lebih terarah dan memudahkan dalam pembahasan sehingga tujuan penelitian akan tercapai. Batasan masalah dalam penelitian ini adalah sebagai berikut:

- Penelitian dilakukan untuk cakupan *computer vision* dalam mengenali objek buah-buahan pada *supermarket*;
- Sistem yang dibangun ini, berbentuk *prototype* yang diperuntukkan bagi konsumen *supermarket* sebagai *user;*
- Sistem ini dihubungkan dengan *database list* harga buah-buahan untuk penghitungan harga buah.

C. Tujuan Penelitian

Tujuan dari penelitian ini adalah membangun sebuah sistem yang memberikan efisiensi bagi perusahaan yang bergerak di bidang bisnis *supermarket* untuk memberikan kemudahan bagi konsumen dalam melakukan *self service*, memperoleh informasi harga buah-buahan dengan memanfaatkan teknologi *computer vision*.

II. KAJIAN LITERATUR

Dalam penelitian ini, menggunakan literatur-literatur yang terkait seputar *computer vision, opencv, histogram of oriented gradient* (*HoG*), dan *k-nearest neighbors* (*k-NN*).

A. Computer Vision

Computer vision, sering disingkat dengan CV, didefinisikan sebagai bidang studi yang berupaya mengembangkan teknik untuk membantu komputer "melihat" dan memahami konten gambar *digital* seperti foto dan *video*. Permasalahan *computer vision* tampak sederhana, namun tetap menjadi masalah yang masih belum dapat terpecahkan berdasarkan pada pemahaman yang terbatas tentang penglihatan biologis dan karena kompleksitas persepsi penglihatan dalam dunia fisik yang dinamis dan hampir tak terbatas.

Computer Vision adalah bidang studi yang berfokus pada masalah membantu komputer untuk melihat [5]. Pada tingkat abstrak, tujuan masalah *computer vision* adalah menggunakan data gambar yang diamati untuk menyimpulkan sesuatu tentang dunia. *Computer vision* sebagai bidang adalah batas intelektual. Seperti batas-batas lainnya, batas tersebut menarik dan tidak terorganisasi, dan sering kali tidak ada otoritas yang dapat diandalkan [6].

Banyak ide berguna yang tidak memiliki landasan teoritis, dan beberapa teori tidak berguna dalam praktiknya. Tujuan dari *Computer vision* adalah untuk memahami konten gambar *digital*. Biasanya, ini melibatkan pengembangan metode yang berupaya mereplikasi kemampuan penglihatan manusia. Memahami konten gambar *digital* mungkin melibatkan penggalian deskripsi dari gambar, yang dapat berupa objek, deskripsi teks, model tiga dimensi.

Sedangkan menurut pandangan lain, *Computer vision* adalah ekstraksi informasi secara otomatis dari gambar. Informasi dapat berarti apa saja mulai dari model 3D, posisi kamera, deteksi dan pengenalan objek hingga pengelompokan dan pencarian konten gambar [7].

Computer vision akan mengolah data berupa gambar dan *video* yang disebut *Image Processing*. *Image Processing* adalah proses membuat gambar baru dari gambar yang ada, biasanya menyederhanakan atau meningkatkan konten dengan cara tertentu. Ini adalah jenis pemrosesan sinyal *digital* dan tidak peduli dengan memahami konten suatu gambar. Contoh *image processing* meliputi :

- Menormalkan sifat fotometrik gambar, seperti kecerahan dan warna;
- Memotong batas gambar, seperti memusatkan objek pada foto;
- Menghapus *noise digital* dari suatu gambar, seperti artefak *digital* dari tingkat cahaya rendah.

Bidang *computer vision* masih memiliki tantangan besar dalam penerapannya. Tujuan dari *computer vision* adalah untuk mengekstraksi informasi yang berguna dari gambar. Ini telah membuktikan tugas yang sangat menantang; hal ini

telah melibatkan ribuan pemikiran cerdas dan kreatif selama empat dekade terakhir.

Meskipun demikian, masih dianggap jauh untuk dapat membangun "mesin penglihatan" yang dapat digunakan untuk tujuan umum. Terdapat permasalahan-permasalahan tingkat tinggi yang memperoleh keuntungan dari penggunaan computer vision pada bidangnya masing-masing [8]:

- *Optical Character Recognition* (*OCR*);
- *Machine inspection;*
- *Retail* (*automated checkouts*);
- *3D model building;*
- *Medical imaging;*
- *Automotive safety;*
- *Motion capture;*
- *Surveillence;*
- *Fingerprint recognition and biometrics.*

Banyak aplikasi *computer vision* yang populer melibatkan upaya untuk mengenali hal-hal dalam foto, sebagai contoh :

- *Object classification;*
- *Object identification;*
- *Object verification;*
- *Object detection;*
- *Object segmentation;*
- *Object recognition.*

B. OpenCV

Open Source Computer Vision Library (*OpenCV*) adalah perpustakaan perangkat lunak yang bersifat *open source* untuk *machine learning* dan *computer vision*. *OpenCV* dibangun untuk menyediakan infrastruktur umum untuk aplikasi *computer vision* dan untuk mempercepat persepsi mesin pada produk komersial. Menjadi produk berlisensi *BSD*, *OpenCV* memudahkan bisnis untuk memanfaatkan dan memodifikasi kode. Perpustakaan memiliki lebih dari 2500 algoritma yang dioptimalkan, yang mencakup serangkaian komprehensif *computer vision* dan algoritma *machine learning* yang mutakhir [9].

OpenCV dapat digunakan untuk mendeteksi dan mengenali bentuk / rupa dari wajah, mengidentifikasi objek, mengklasifikasikan tindakan manusia dalam video, melacak gerakan benda, mengekstraksi model objek 3D, menghasilkan awan titik 3D dari kamera *stereo*, menggabungkan beberapa gambar menjadi satu kesatuan untuk menghasilkan resolusi tinggi gambar seluruh adegan, temukan gambar serupa dari basis data gambar, hapus mata merah dari gambar yang diambil menggunakan lampu flash, mengikuti gerakan mata, mengenali pemandangan, membuat lapisan *augmented reality*, dan lainnya.

OpenCV memiliki antarmuka C++, *Python*, *Java* dan *MATLAB* dan mendukung *Windows*, *Linux*, *Android* dan *Mac OS*. *OpenCV* sebagian besar condong ke aplikasi penglihatan *realtime* dan memanfaatkan instruksi *MMX* dan *SSE* saat *tersedia*. Dengan *OpenCV*, dapat dilakukan pengolahan gambar objek buah-buahan untuk *supermarket*.

C. Histogram of Oriented Gradients (HoG)

HoG atau *Histogram of Oriented Gradients* adalah *feature descriptor* yang sering digunakan untuk mengekstrak fitur dari data gambar. Ini banyak digunakan dalam tugas *computer vision* untuk deteksi objek [10]. *Feature descriptor* adalah representasi gambar atau *patch* gambar yang menyederhanakan gambar dengan mengekstraksi informasi yang berguna dan membuang informasi asing.

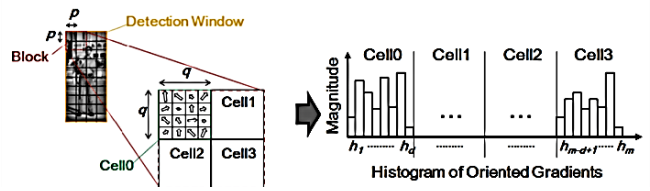
Dalam *HoG feature descriptor*, distribusi (*histogram*) arah gradien (gradien berorientasi) digunakan sebagai fitur. Gradien (turunan x dan y) dari suatu gambar berguna karena nilai gradien di sekitar tepi dan sudut adalah besar (daerah dengan perubahan intensitas mendadak). Selain itu, daerah tepi dan sudut mengemas lebih banyak informasi tentang bentuk objek daripada daerah datar dari suatu gambar [11]. Beberapa aspek penting *HOG* yang membuatnya berbeda dari deskriptor fitur lainnya:

- *Descriptor HOG* berfokus pada struktur atau bentuk objek. Ini dilakukan dengan mengekstraksi gradien dan orientasi;
- Selain itu, orientasi ini dihitung dalam porsi 'terlokalisasi'. Ini berarti bahwa gambar lengkap dipecah menjadi daerah yang lebih kecil dan untuk setiap daerah, gradien dan orientasi dihitung;
- Akhirnya *HoG* akan menghasilkan *histogram* untuk masing-masing wilayah ini secara terpisah. *Histogram* dibuat menggunakan gradien dan orientasi nilai piksel.

Secara keseluruhan, dapat didefinisikan sebagai berikut: *feature descriptor HoG* menghitung kemunculan orientasi gradien di bagian gambar yang dilokalkan. *Feature descriptor HoG* dapat digunakan untuk mengekstraksi fitur berupa bentuk objek dan juga warna objek pada gambar. Keduanya dapat dilakukan dengan cara kerja yang serupa:

- Langkah pertama adalah melakukan *preprocessing* pada data gambar. *Preprocessing* data dapat dilakukan dengan cara *resize* gambar menjadi ukuran piksel yang lebih kecil dari gambar awal. Misalkan kita memiliki gambar awal dengan ukuran 180x280 piksel, maka kemudian dilakukan *resize* menjadi gambar dengan ukuran 64x128 piksel;
- Langkah kedua adalah melakukan kelkulasi gradien pada setiap piksel dalam gambar. Langkah ini akan menghasilkan dua buah matriks, satu matriks menyimpan gradien arah sumbu x dan satu matriks menyimpan gradien arah sumbu y;
- Langkah ketiga adalah menghitung *magnitude* dan orientasi gradien dari matriks yang diperoleh sebelumnya. Cara kerja *HoG* terdapat pada Gambar 1.

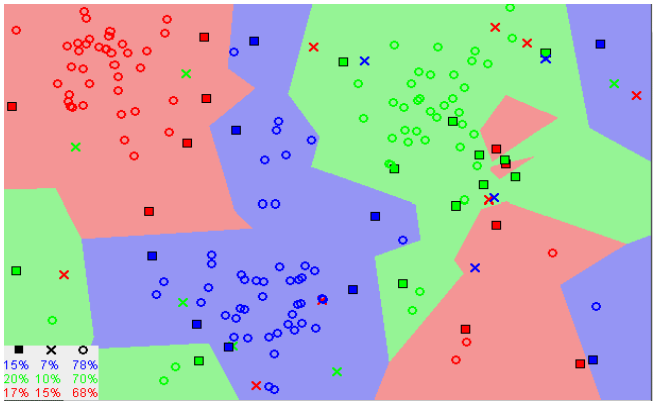
p-ISSN : 2443-2210
e-ISSN : 2443-2229



Gambar 1. Cara kerja *HoG*

D. K-Nearest Neighbors (knn)

Algoritma *K-Nearest Neighbors* adalah algoritma pembelajaran mesin sederhana, mudah diterapkan yang dapat digunakan untuk memecahkan masalah klasifikasi dan regresi. Algoritma ini bekerja untuk pembelajaran dengan label atau *supervised*. Algoritma *supervised learning* adalah algoritma yang bergantung pada data *input* berlabel untuk mempelajari fungsi yang menghasilkan *output* yang sesuai ketika diberi data baru tanpa label. Algoritma *knn* mengasumsikan bahwa hal serupa ada dalam jarak dekat. Dengan kata lain, hal-hal serupa dekat satu sama lain [12].



Gambar 2. Contoh K-Nearest Neighbors

Seperti pada contoh Gambar 2, bahwa sebagian besar waktu, titik data yang serupa dekat satu sama lain. Algoritma *k-NN* bergantung pada asumsi tersebut, *knn* menangkap gagasan kesamaan (kadang-kadang disebut jarak, kedekatan, atau kedekatan). Cara yang digunakan adalah menghitung jarak kedekatan antara suatu titik dengan titik lainnya.

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
knn_clf = KNeighborsClassifier(n_jobs=-1, weights='distance', n_neighbors=11)
knn_clf.fit(X_trainftr, y_train)
```

Gambar 3. Algoritma *k-NN*

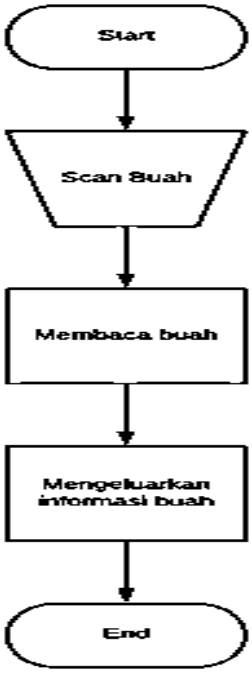
Algoritma *k-NN* seperti pada Gambar 3 digunakan untuk mengklasifikasikan objek baru berdasarkan atribut dan *training sample*. Dalam implementasinya, algoritma ini memiliki keunggulan:

- Algoritma ini termasuk mudah digunakan;
- Tidak perlu membangun *model*, *tuning parameter*;
- Memiliki banyak kegunaan, bisa untuk klasifikasi atau regresi.

III. PERANCANGAN

Berikut pada Gambar 4 adalah rancangan *flowchart* yang akan digunakan. Pelanggan akan meletakan buah pada timbangan, lalu kamera akan mengambil gambar buah yang diletakkan oleh pelanggan tersebut. Gambar buah yang diambil oleh kamera akan menjadi *input* bagi sistem. Kemudian sistem akan mengenali jenis buah tersebut dan mengeluarkan informasi yang berisikan data nama buah dan harga buah berdasarkan satuan berat kilogram.

A. Flowchart



Gambar 4. *Flowchart* penelitian yang dilakukan

IV. METODOLOGI

Dalam penelitian ini, *tools* yang digunakan adalah *OpenCV*, *HoG*, dan *k-NN*. Tugas dilakukan dengan mengikuti pedoman alur kerja / proses pembuatan *model* pembelajaran *fruit detection* seperti pada Gambar 5 berikut.

Gambar 5. Alur kerja *Fruit Detection*

Langkah pertama yang dilakukan adalah memasukan *dataset* sebanyak 9.695 gambar sebagai *input* yang terdiri atas:

- Buah apel (3.110 gambar);
- Buah pisang (2.838 gambar);
- Buah lemon (3.747 gambar).

Langkah kedua, dilakukan pemisahan (*splitting*) data menjadi data untuk *training* sebanyak 70% dan data untuk *testing* sebanyak 30%.

```
# Split data training dan Testing 70% training, 30% testing
X_train, X_test, y_train, y_test = train_test_split(X_Colordes, y_Colordes, tes
t_size=0.30, random_state=42)
```

Gambar 6. Data dipisah untuk *training* dan *testing*

Setelah memisahkan data *training* dan *testing* pada Gambar 6, dilakukan *preprocessing* pada data gambar. Hal ini dilakukan dengan *OpenCV* di mana kegiatan yang dilakukan adalah sebagai berikut:

- *Read image file;*
- Deteksi objek pada gambar;
- Konversi warna;
- *Crop;*
- *Resize.*

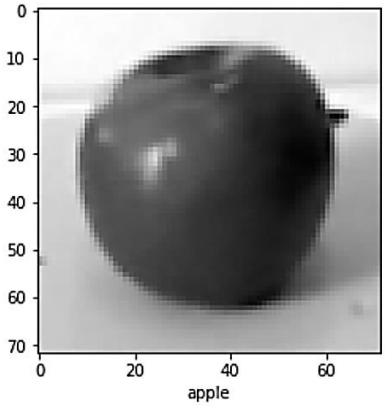
Namun ada perbedaan dalam *preprocessing* antara gambar yang nantinya akan dilakukan ekstraksi fitur berdasarkan bentuk seperti terlihat pada Gambar 8 serta gambar yang dilakukan ekstraksi fitur berdasarkan warna [13]. Bagi gambar yang akan diekstraksi fitur berupa bentuk, dilakukan *preprocessing* berupa merubah warna objek menjadi *grayscale*. Sedangkan bagi gambar yang akan diekstraksi fitur berupa warna, dilakukan *preprocessing* berupa merubah warna *RGB* menjadi *HSV* seperti pada Gambar 7. Hasil dari *preprocessing* merubah warna *RGB* menjadi *HSV* dapat dilihat pada Gambar 9.

```
In [13]: # Preprocessing mengubah rgb ke hsv
         X_trainp=preprocessing2(X_train)
         X_testp=preprocessing2(X_test)
```

Gambar 7. *Preprocessing* untuk fitur warna

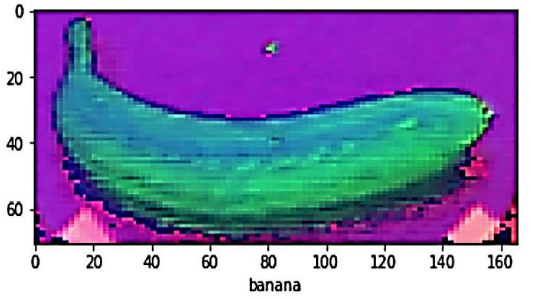
```
In [13]: #Preprocessing conversi gambar ke gray
         X_trainp=preprocessing1(X_train)
         X_testp=preprocessing1(X_test)
```

```
In [14]: # Cek data Preprocessing 72 x 72, lihat
         img=X_trainp[1]
         y = y_train[1]
         plt.imshow(img,cmap="gray")
         plt.xlabel(y)
         plt.show()
```



Gambar 8. *Preprocesing* untuk fitur bentuk

```
In [14]: img=X_trainp[1]
         y = y_train[1]
         plt.imshow(img)
         plt.xlabel(y)
         plt.show()
```

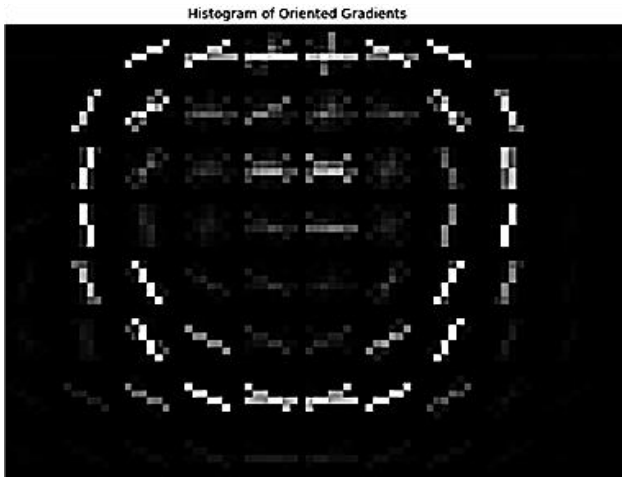


Gambar 9. Hasil *preprocessing rgb* menjadi *hsv*

Data gambar yang sudah dilakukan *preprocessing* kemudian akan dilakukan ekstraksi fitur dengan *HoG* seperti pada Gambar 10 dan akan memberikan hasil akhir seperti pada Gambar 11. Fitur-fitur yang diekstraksi adalah bentuk objek pada gambar dan warna dari objek pada gambar.

```
#Feature Extraction mengambil data2 yang penting saja dari 5184 feature menjadi
128 pixel saja
X_trainftr=featureExtraction1(X_trainp)
X_testftr=featureExtraction1(X_testp)
```

Gambar 10. Ekstraksi fitur dengan *HoG*



Gambar 11. Hasil ekstraksi fitur

Langkah selanjutnya yang dilakukan setelah mengekstraksi fitur dari gambar adalah melakukan klasifikasi berdasarkan jenis buah (apel, pisang, lemon). Klasifikasi seperti pada Gambar 12 dilakukan dengan menggunakan algoritma *k-NN* (*K-nearest neighbors*).

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
knn_clf = KNeighborsClassifier(n_jobs=-1, weights='distance', n_neighbors=11)
knn_clf.fit(X_trainftr, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=-1, n_neighbors=11, p=2,
                     weights='distance')
```

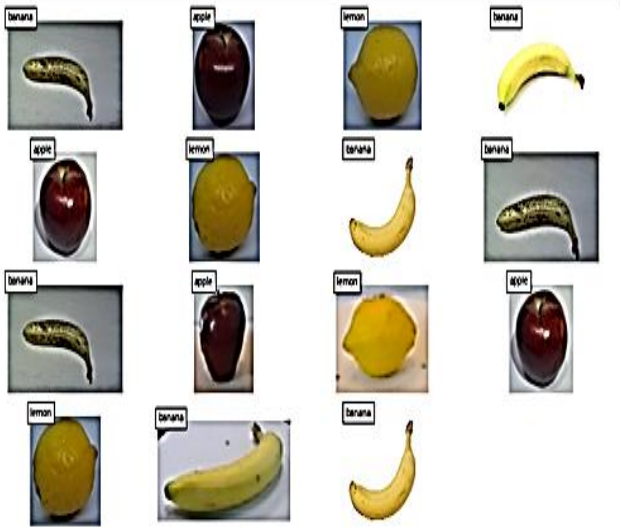
Gambar 12. Klasifikasi dengan *k-NN*

Selanjutnya dilakukan *testing* dengan data yang sudah dilakukan ekstraksi fitur bentuk dan fitur warna. Hasil *testing* Gambar 13 menunjukkan bahwa tingkat akurasi bagi fitur bentuk adalah sebesar 99,8% dan bagi fitur warna adalah sebesar 99,7%. Langkah berikutnya yang dilakukan adalah melakukan uji coba dengan data baru di luar *dataset* sebelumnya. Tujuannya adalah untuk mengetahui apakah prediksi buah dapat dilakukan dengan tepat seperti menggunakan *dataset* pada tahap sebelumnya. Jika hasil pengujian sukses membaca buah pada gambar baru, maka sistem yang dibentuk berhasil.

Gambar buah yang diperoleh dengan hasil *capture* dari kamera akan dilakukan *preprocessing* dan ekstraksi fitur seperti pada Gambar 14. Kemudian gambar yang diperoleh dijadikan sebagai *input* bagi sistem pengenalan buah-buahan. Sistem akan mengenali buah tersebut dan mencari informasi nilai harga dari buah tersebut. Untuk harga yang dimunculkan oleh sistem, tersedia dalam *format* daftar harga per satuan kilogram menggunakan sampel buatan terlebih dahulu untuk pengujian, bukan menggunakan data harga sebenarnya. Hasil keluaran dari sistem dapat dilihat pada Gambar 15.

```
slice = len(path)

plt.figure(figsize=(16,8))
for i in range(slice):
    plt.subplot(4, 4, i+1)
    plt.imshow(buah[i], interpolation='nearest')
    plt.text(0, 0, y_knn_pred[i], color='black',
             bbox=dict(facecolor='white', alpha=1))
    plt.axis('off')
```



Gambar 13. *Testing* data gambar

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
from skimage.transform import rescale, resize

cap = cv2.VideoCapture(0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Our operations on the frame come here
    buah = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    # Display the resulting frame
    cv2.imshow('frame',buah)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()


#img=resize(gray, (64, 128),anti_aliasing=True) # test preprocessing
#plt.imshow(img, cmap='gray')
#plt.show()

# simpan gambar
cv2.imwrite('./uji_buah.jpg',frame)
```

Gambar 14. *Input* gambar buah melalui kamera

```
pred = knn_clf.predict([imgftr])
```

```
if pred[0] == apple:
    print("ini buah", pred)
    print("Harga 5000/Kg")
elif pred[0] == lemon:
    print("ini buah", pred)
    print("Harga 4500/Kg")
else:
    print("ini buah",pred)
    print("Harga 3500/Kg")
```

```
ini buah ['banana']
Harga 3500/Kg
```

Gambar 15. Hasil prediksi informasi buah dan harganya

V. RISIKO

Dalam perancangan prototipe sistem pengenalan buah-buahan ini, selain memastikan sistem yang dibangun dapat digunakan sebagaimana mestinya, kita tidak boleh menutup mata terhadap hal-hal yang dapat saja muncul sebagai ancaman yang dapat menghambat kinerja dari sistem tersebut. Ancaman-ancaman yang bisa saja muncul dianggap sebagai risiko dan harus ada tindakan yang dilakukan untuk menanggulanginya apabila sistem yang dibangun ingin dapat diaplikasikan dengan baik dan tanpa kendala. Dalam pembentukan sistem pengenalan buah, terdapat berbagai potensi ancaman dapat terjadi pada setiap domain manajemen risiko terkait implementasi Teknologi Informasi. Domain-domain terkait dengan ancaman merupakan bagian alur kerja dari sistem yang meliputi [14]:

- *User domain* - Ketidak tahuan *user* dalam penggunaan kamera atau program dari sistem pengenalan buah-buahan;
- *Workstation domain* - Terdapat *virus* atau program *error* sehingga sistem pengenalan buah tidak dapat digunakan;
- *LAN domain* - Terdapat masalah pada jaringan *local* sehingga aplikasi tidak dapat digunakan untuk mengenali buah-buahan;
- *LAN to WAN domain* - Terdapat masalah dalam akses program dan data program (daftar nama buah dan harganya), sehingga saat dijalankan, program tidak bisa mengeluarkan *output*;
- *Remote access domain* - Tidak bisa mengeluarkan *output* karena terdapat masalah pada jaringan *local* ke pusat (data nama buah dan informasi harga disimpan pada kantor pusat);
- *WAN domain* - Terdapat masalah pada koneksi dari penyedia layanan internet (*ISP*);
- *System application domain* - Terdapat masalah pada aplikasi atau data sehingga aplikasi tidak berjalan lancar.

Ancaman-ancaman tersebut adalah risiko yang mungkin dapat terjadi pada saat sistem pengenalan buah digunakan oleh *user*. Untuk kemungkinan langkah mitigasinya, bisa dilakukan :

- Membuat *tutorial* penggunaan sistem yang mudah dimengerti oleh pelanggan;
- Menggunakan perangkat lunak *antivirus* berbayar;
- Membuat *alert system* jika jaringan terputus agar dapat langsung ditangani;
- Menyediakan alternatif penyedia jasa layanan internet (*ISP*) lebih dari 1;
- Membuat *time scheduler* untuk dapat melakukan *update database* harga buah secara otomatis.

VI. KESIMPULAN

Sesuai dengan hasil seperti pada Gambar 15, dapat disimpulkan bahwa sistem yang dibentuk mampu melakukan prediksi hasil informasi nama buah dan harga per satuan kilogram berdasarkan data baru yang diambil dari kamera. Ini adalah *prototype* yang sebagian besar mewakili rencana awal yang diharapkan, yaitu sistem *price checker* yang mampu menampilkan informasi mengenai buah dan harganya. Perlu juga diperhatikan dengan adanya sistem ini, maka akan muncul juga berbagai risiko terkait yang perlu dibuat perencanaan mitigasinya, sehingga dapat meminimalkan dampak yang ditimbulkan bagi sistem.

DAFTAR PUSTAKA

[1] Olyver Wyman. (2018) The Future Supermarket: How Digital Operations Will Enable a Winning Customer Experience, At Much Lower Cost. [Online]. Tersedia: *https://www.oliverwyman.com/our-expertise/insights/2018/feb/retail-consumer-journal-vol-6/the-future-supermarket.html*

[2] (2019) From smart carts to computer vision, checkouts innovation are multiplying. [Online]. Tersedia: http://www.grocerydive.com/news/from-smart-carts-to-computer-vision-checkout-innovations-are-multiplying/563425/

[3] (2019) 4 Computer Vision Technologies For Brick and Mortar Stores. [Online]. Tersedia: https://www.clarifai.com/blog/4-computer-vision-technologies-for-brick-and-mortar-stores

[4] (2001) Veggie Vision. [Online]. Tersedia: http://www.supermarketnews.com/archive/veggie-vision

[5] S. Prince, *Computer Vision: Models Learning and Inference*, Cambridge: University Press, 2012.

[6] D. A. Forsyth & J. Ponce, *Computer Vision – A Modern Approach*, 2nd ed., Pitman, 2012.

[7] J. E. Solem, *Programming Computer Vision with Python*, Cambridge: O'Reilly Media, 2012.

[8] R. Szeliski, *Computer Vision Algorithms and Applications*, London, New York: Springer, 2011.

[9] (2020) The OpenCV website. [Online]. Tersedia: http://www.opencv.org/

[10] (2019) Feature Descriptor. [Online]. Tersedia: http://www.analyticsvidhya.com/blog/2019/09/feature-engineering-image-introduction-hog-feature-descriptor/

[11] (2016) Histogram of Oriented Gradients. [Online]. Tersedia: http://www.learnopencv.com/histogram-of-oriented-gradients/

[12] (2018) K-Nearest Neighbors. [Online]. Tersedia: http://www.towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neoghbors-algorithm-6a6e71d01761

[13] R. Bolle, J. Connell, N. Haas, R. Mohan &G. Taubin,"VeggieVision: A Produce Recognition System" *IEEE Workshop on Automatic Identification Advanced Technologies*., WAIAT-97, , November. 1997, pp. 35-38.

[14] (2018) The Seven Domain of a Typical IT Infrastructure. [Online]. Tersedia: http://www.sis.binus.ac.id/2018/01/15/the-seven-domain-of-a-typical-it-infrastructure/