BAB V

ANALISIS DAN HASIL PENELITIAN PELACAK TANGAN

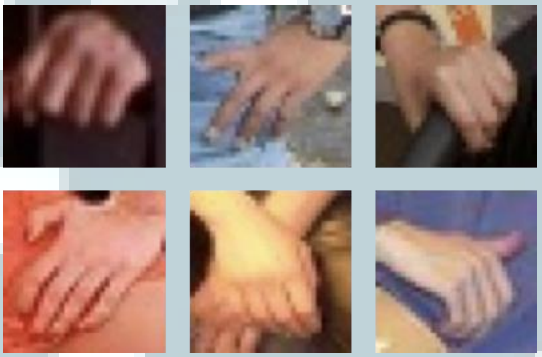
5.2 Pelacak Tangan

Pembahasan pelacak tangan pada bab ini akan dibagi menjadi 5 bagian yaitu, (1) deskripsi data (2) 10-*fold cross validation* model deteksi tangan, (3) pengujian akurasi model deteksi tangan, (4) pengujian akurasi model pelacak tangan, (5) hasil lokalisasi model pelacak tangan pada video.

5.2.1 Deskripsi Data

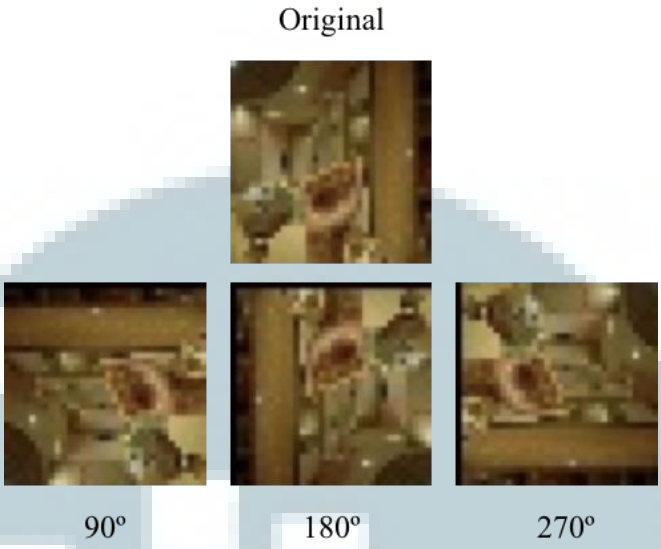
Data yang digunakan untuk melatih kembali model CNN yang dipilih memiliki performa terbaik adalah data yang telah dikumpulkan sebelumnya dan data yang diperoleh dari penelitian Mittal, Zisserman, & Torr (2011) yang berasal dari *Oxford University*. Pada dataset Oxford, terdapat 3 subset yaitu *training*, *validation* dan *test*. Subset yang digunakan pada tahap ini hanya *training set* dengan jumlah 4070 gambar contoh data dapat dilihat pada gambar 5.1. Karena gambar yang ada pada dataset Oxford merupakan gambar yang memiliki multi objek, setiap tangan pada masing-masing gambar akan di *crop* menggunakan anotasi yang telah diberikan oleh dataset Oxford. Hasil dari *cropping* dapat dilihat pada gambar 5.2.

Gambar 5.1 Contoh data pada *training set* dataset Oxford



Gambar 5.2 Contoh data yang sudah di *crop* pada *training set* dataset Oxford

Selanjutnya dengan data yang telah dikumpulkan sebelumnya yaitu data NUS II untuk melatih model, pada tahap ini data tersebut akan digunakan untuk augmentasi dimana augmentasi hanya merotasi gambar 90º, 180º, dan 270º. Beberapa sampel hasil augmentasi dapat dilihat pada gambar 5.3.



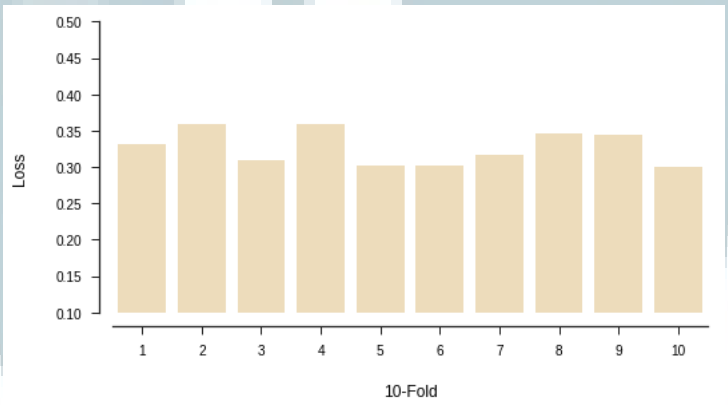
Gambar 5.3 Contoh hasil augmentasi pada data yang ada pada dataset NUS II

5.2.2 10-*Fold Cross Validation* Model Deteksi Tangan

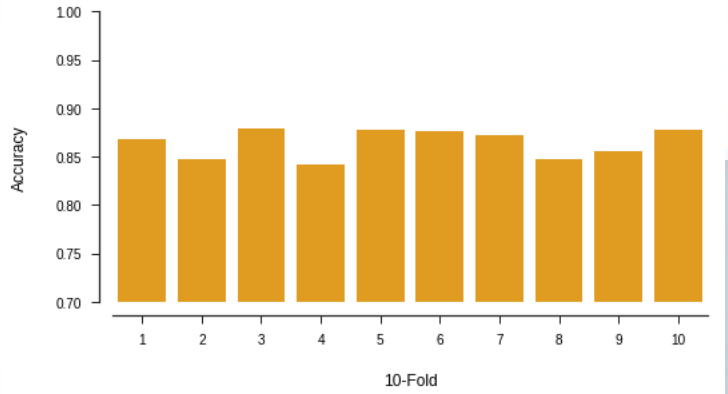
Hasil dari melakukan pelatihan model deteksi tangan dengan dataset NUS I, NUS II dan Oxford, menunjukkan hasil peforma yang cukup baik dimana dapat dilihat pada gambar 5.4. Pada gambar tersebut, *loss* dari seluruh *fold* memiliki nilai yang kecil dimana *loss* terbesar adalah 0.35 pada *fold* ke-2 dan ke-4 dan yang terkecil adalah 0.29 pada *fold* ke-10.

Untuk melihat adanya hubungan antara nilai *loss* dengan besarnya akurasi yang dicapai, diperlukan analisa besarnya akurasi yang dicapai setiap *fold* pada pelatihan model. Gambar 5.5 merupakan gambar yang menunjukkan besarnya akurasi yang dicapai setiap *fold* terhadap dataset yang telah dipecah untuk melakukan validasi. Dataset tersebut terdiri dari data-data pada dataset NUS I, NUS II dan Oxford. Akurasi tertinggi adalah 87% yang diperoleh oleh *fold* ke-5 dan ke-10. Akurasi terendah adalah 84% yang diperoleh oleh *fold* ke-2, ke-4, dan ke-8.

Akurasi tertinggi dan *loss* terendah pada hasil pelatihan model ada pada *fold* ke-10 dan akurasi terendah dan *loss* tertinggi terdapat pada *fold* ke-2 dan ke-4. Dari hasil tersebut dapat disimpulkan bahwa, jika nilai *loss* semakin mendekati *local minimum* atau semakin rendah terhadap hasil prediksi dengan *true label*, maka akurasi yang diperoleh semakin tinggi. Kemudian jika kedua *barchart* dibandingkan satu sama lain, setiap *bar* pada *fold* tertentu akan menimbulkan efek bertolak belakang, semakin tinggi *loss* semakin rendah akurasi dan sebaliknya.

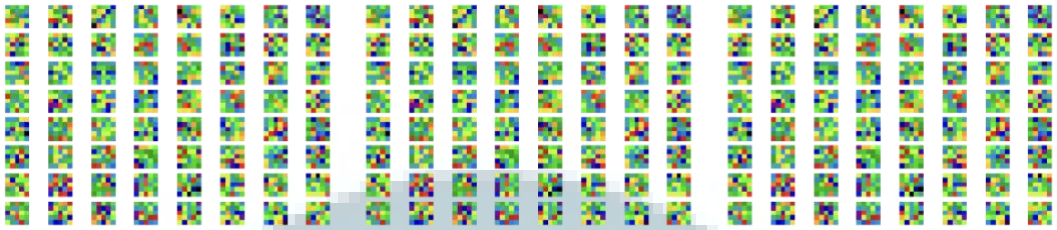


Gambar 5.4 Barchart *loss* pada 10-*cross validation*

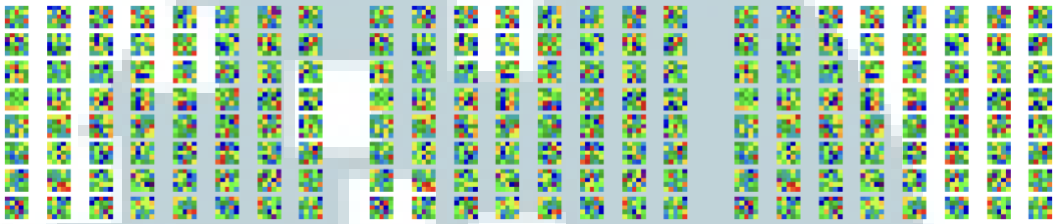


Gambar 5.5 Barchart akurasi yang diperoleh pada 10-*cross validation*

Visualisasi *weight* pada layer konvolusi ke-1 dapat dilihat pada gambar 5.6 dan pada layer konvolusi ke-2 dapat dilihat pada gambar 5.7.



Gambar 5.6 Visualiasi 3 channel nilai weight pada layer ke-1 fold indeks ke-1



Gambar 5.7 Visualiasi 3 channel nilai weight pada layer ke-2 fold indeks ke-1

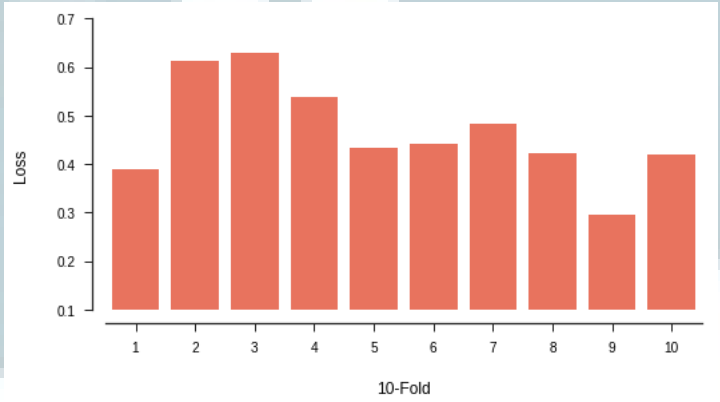
5.2.3 Pengujian Akurasi Model Deteksi Tangan

Pada pengujian akurasi model deteksi yang dilakukan untuk membuat model pelacak tangan ini, model di uji dengan data yang sama untuk menguji model deteksi tangan yaitu dataset NUS I. Hasil yang didapatkan berbeda dengan pelatihan model deteksi tangan sebelumnya yang tidak menggunakan teknik augmentasi data. Model deteksi yang telah dilatih dapat mencapai performa yang lebih baik untuk melakukan klasifikasi terhadap data NUS I.

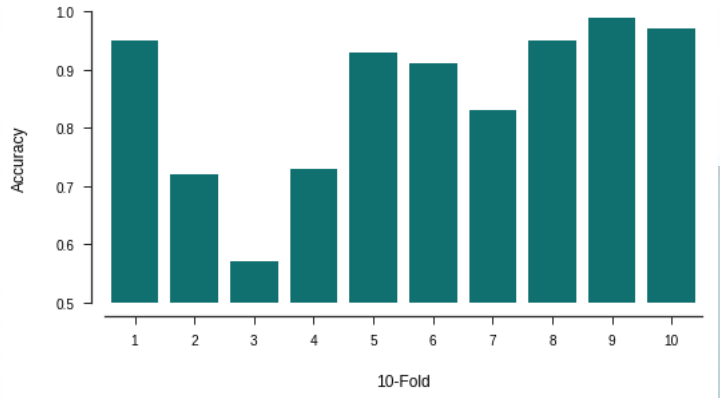
Pada gambar 5.6 menunjukkan hasil akhir *loss* yang dihitung berdasarkan rata-rata dari *loss* hasil prediksi setiap data yang ada pada masing-masing *fold*. Nilai *loss* terbesar terdapat pada *fold* ke-3 yaitu 0.62 dan nilai *loss* terendah terdapat pada *fold* ke-9 yaitu 0.29. Hasil yang didapat dari pengujian akurasi model ini cukup fluktuatif dibandingkan dengan saat validasi.

Pada gambar 5.7 menunjukkan akurasi yang diperoleh saat pengujian akurasi model. Model yang telah dilatih memperoleh akurasi yang cukup tinggi di

beberapa *fold*. Akurasi tertinggi yaitu 99% pada *fold* ke-9 dan akurasi terendah yaitu 57% pada *fold* ke-3. Hal ini dibuktikan dari nilai *loss* yang diperoleh pada *fold* ke-9 dan *fold* ke-3 dimana nilai yang satu paling rendah dan yang satunya lagi memiliki nilai paling tinggi. Hasil yang baik ini diperoleh karena adanya augmentasi data yang dilakukan pada dataset NUS II sehingga memperkaya varians dari masing-masing postur tangan. Terbukti dengan teknik ini dapat memperkaya pengetahuan dari model yang dilatih.



Gambar 5.6 Barchart *loss* pengujian akurasi pada 10-*cross validation*



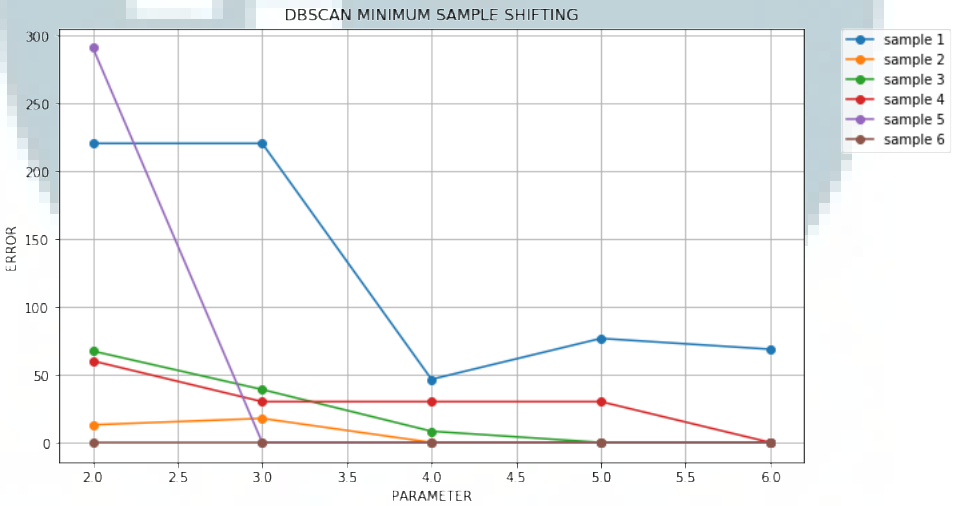
Gambar 5.7 Barchart hasil pengujian akurasi pada 10-*cross validation*

5.2.4 Pengujian Akurasi Model Pelacak Tangan

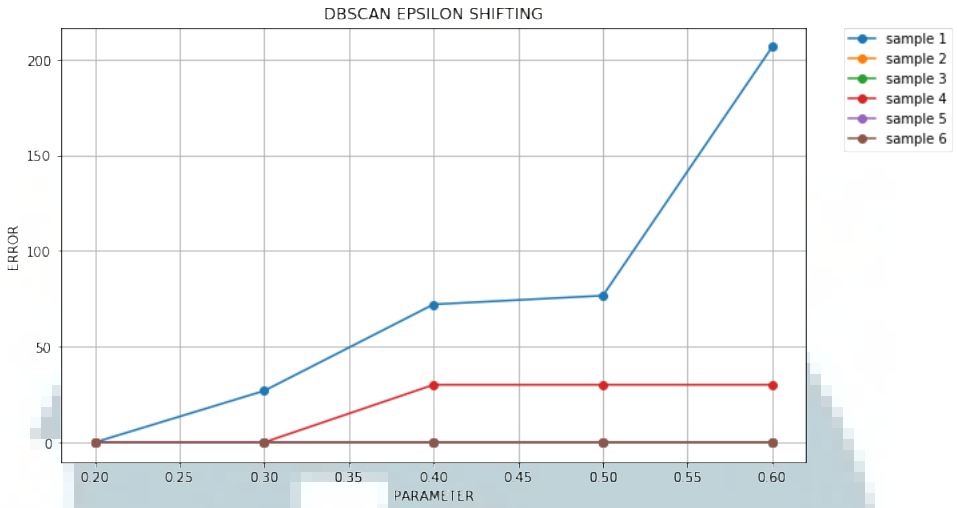
Pengujian akurasi model pelacak tangan dilakukan dengan cara menghitung *error rate* yang dihitung dengan menggunakan *euclidean distance* dan *distance* antara titik *x* dan *y*. Perhitungan *error rate* dilakukan untuk seluruh uji coba menggunakan algoritma *clustering* yaiut *k-means*, *dbscan,* dan *mean shift*. Masing-masing uji coba parameter untuk setiap algoritma *clustering* juga dihitung seberapa besar *error rate* yang didapati. Hal ini berguna untuk membandingkan algoritma *clustering* mana sebagai pendukung terbaik dan melihat dampak yang ditimbulkan atas perubahan parameter terhadap lokalisasi tangan. Dari 5 video yang dibuat untuk menguji model pelacak tangan, hasil yang diperoleh dapat dilihat pada gambar 5.8, 5.9, 5.10 dan 5.11. Gambar-gambar tersebut merupakan hasil *distance error* yang diperoleh dari *euclidean distance* antara *centroid* dengan rata-rata dari *cluster* modus.

Dari keempat gambar ini, dapat dilihat dampak yang terjadi jika nilai parameter berubah secara inkremental terhadap nilai *distance error* pada 5 buah *sample* video untuk menguji model pelacak tangan. Pada gambar 5.8, menunjukkan hasil *distance error* yang dicapai berdasarkan perubahan parameter *minimum sample* pada algoritma *dbscan*. Kemudian pada gambar 5.9, menunjukkan hasil *distance error* yang dicapai berdasarkan perubahan nilai parameter *epsilon* pada algortima *dbscan* dan sama halnya untuk gambar 5.10 dan 5.11 dimana menunjukkan hasil *distance error* berdasrkan besarnya *k* pada *k-means* dan *quantile* pada *meanshift*.

Pada grafik dalam masing-masing gambar dapat ditarik kesimpulan, dimana pada gambar 5.8 dengan nilai *epsilon* konstan 0.5, menggunakan *minimum sample* sebanyak 4 terbukti dari 5 sample memiliki *distance error* paling rendah dibandingkan dengan yang lainnya. Pada gambar 5.9 dimana merupakan kebalikan dari gambar sebelumnya yakni konstan berubah ke *minimum sample* dimana nilainya adalah 0.5. Dengan nilai *minimum sample* 0.5 sebagai konstan terbukti dengan adanya perubahan nilai parameter pada *epsilon* lebih stabil dan pada gambar tersebut 4 buah sample tidak memiliki *distance error* dari *centroid*.



Gambar 5.8 *Distance error* berdasarkan *euclidean distance* pada *DBSCAN* dengan perubahan parameter *minimum* sample

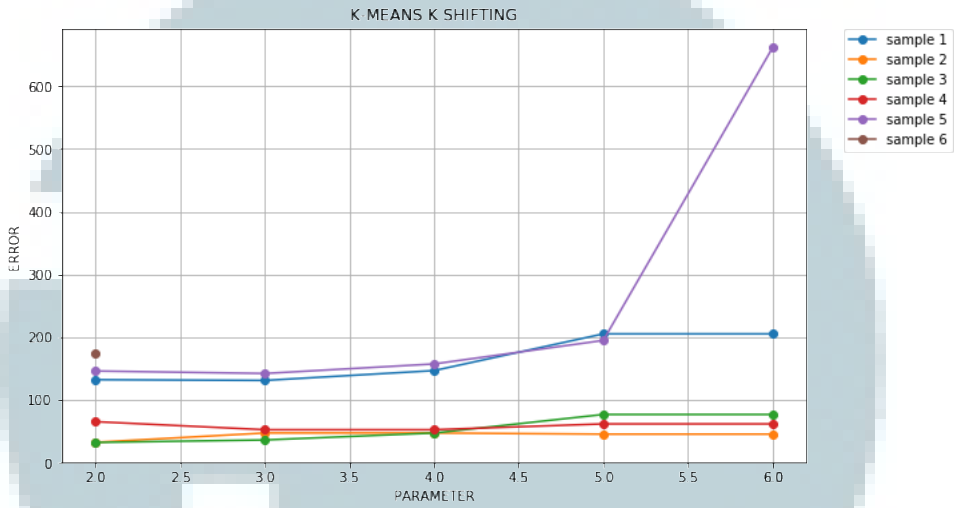


Gambar 5.9 *Distance error* berdasarkan *euclidean distance* pada *DBSCAN* dengan perubahan parameter *epsilon*

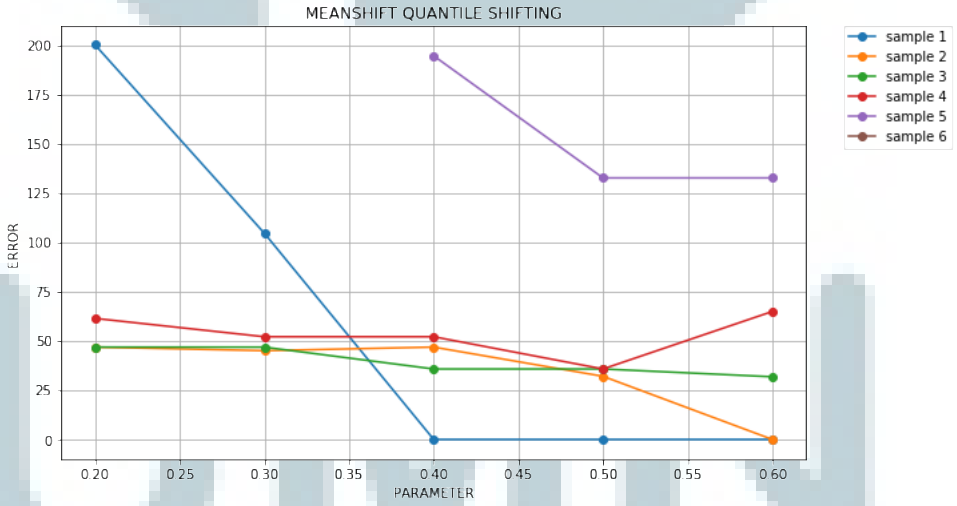
Penggunaan algoritma *k-means* memberikan hasil cukup baik dimana *distance error* cukup rendah. Namun, jika dilihat pada gambar 5.10, bilamana *k* semakin besar *distance error* akan semakin besar dan apabila terlalu kecil *distance error* juga dapat bertambah besar. Pada gambar 5.10 dapat dilihat pada sample ke-6 dimana pada saat pengujian kumpulan *window* yang terdeteksi memiliki jumlah yang sedikit sehingga *k-means* tidak mampu untuk melakukan *clustering*.

Berbeda hasil performa dengan *k-means*, *meanshift* cenderung lebih fluktuatif terhadap seluruh sample (gambar 5.11). Nilai *quantile* terbaik adalah 0.5 dimana secara rata-rata menghasilkan *distance error* yang paling rendah. Namun pada sample ke-5 dengan menggunakan nilai *quantile* 0.2 dan 0.3, *meanshift* tidak mampu melakukan *clustering* terhadap sample. Begitu juga dengan sample ke-6 dimana seluruh nilai *quantile* gagal melakukan *clustering*. Karena *sliding window* dapat mendeteksi tangan dengan tepat dan kumpulan *window* terdeteksi menjadi sedikit, algoritma *meanshift* dirasa kurang baik untuk melakukan *clustering*

terhadap data yang sangat kecil. Akan tetapi, dari seluruh permasalahan tidak dapatnya sebuah algoritma untuk melakukan *clustering*, penggunaan *centroid* sebagai koordinat lokalisasi dapat digunakan untuk alternatif.



Gambar 5.10 *Distance error* berdasarkan *euclidean distance pada k-means* dengan perubahan parameter *k*

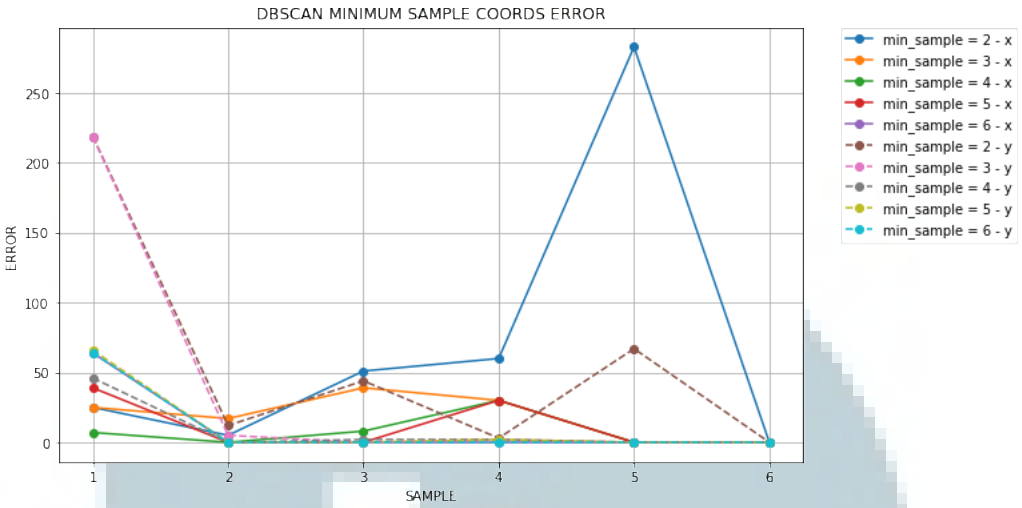


Gambar 5.11 *Distance error* berdasarkan *euclidean distance* pada *mean shift* dengan perubahan parameter *quantile*

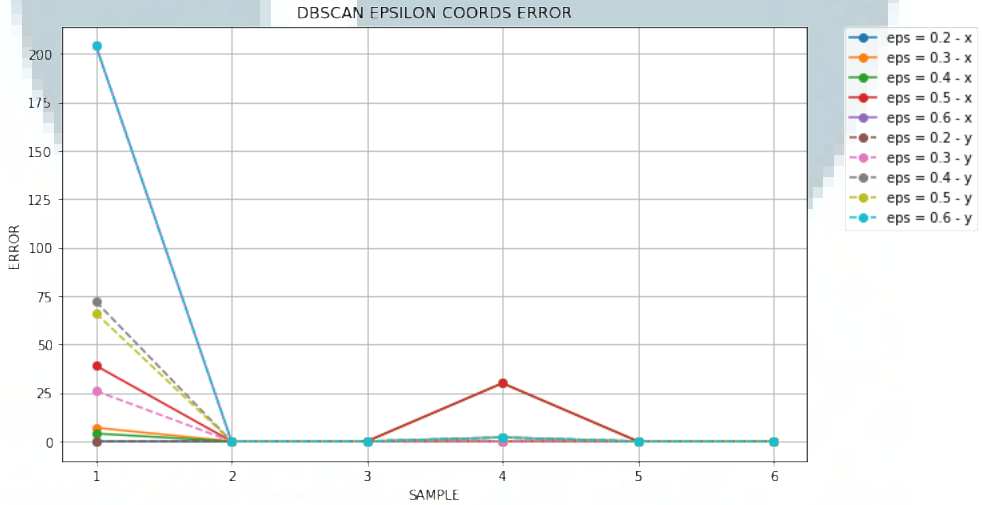
Perhitungan *distance error* juga dihitung tidak hanya dari *Euclidean distance* tetapi juga dari jarak antara titik *x* dengan *xi* dan *y* dengan *yi* dimana *xi* dan

yi adalah titik dari *centroid*. Perhitungan *distance error* ini dikarenakan untuk mengetahui *error* terbesar antara *error* secara vertikal atau horizontal. Pada gambar 5.12, 5.13, 5.14 dan 5.15. Pada gambar-gambar tersebut yang dapat dilihat adalah *distance error* titik *x* dan titik *y* dimana titik estimasi *x* dan *y* juga didapat dari rata-rata *cluster* modus sehingga tetap menggunakan algoritma *clustering* dengan parameter tertentu. *Distance error* dihitung terhadap seluruh sample yang telah dibuat.

Pada gambar 5.12, merupakan grafik hasil *distance error x* dan *y* dimana perubahan parameter ada pada *minimum sample*. Jumlah *minimum sample* terbaik adalah 4 sama seperti evaluasi yang dilakukan pada gambar 5.8. Gambar 5.13 menunjukkan hasil *distance error* dimana perubahan parameter terjadi pada *epsilon*. Pada gambar tersebut seluruh hasil tidak terlalu fluktuatif dan besarnya nilai *epsilon* memperendah *distance error*. Dari kedua gambar tersebut *dbscan* memberikan performa yang cukup baik untuk kumpulan data yang sedikit dikarenakan perubahan parameter tidak membuat batasan terhadap model untuk melakukan *clustering*.



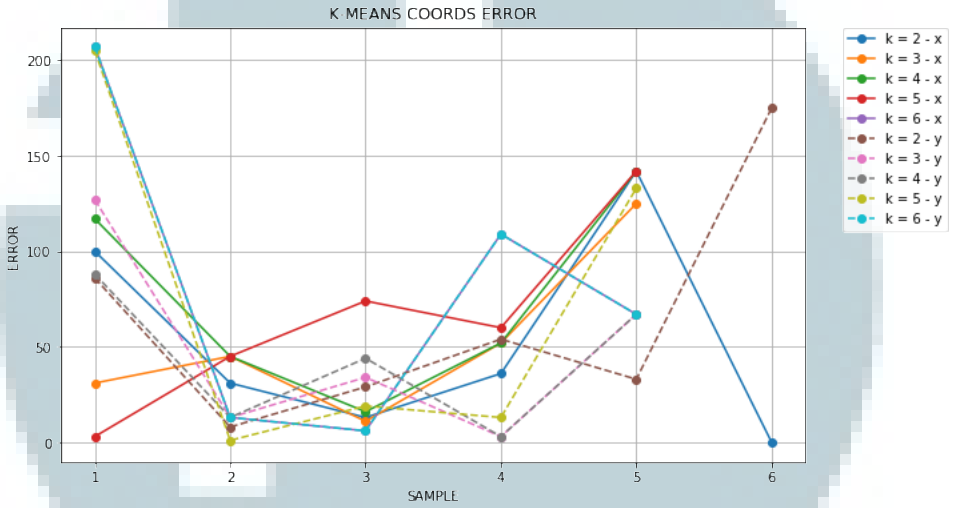
Gambar 5.12 *Distance error* berdasarkan *coordinates distance* pada *DBSCAN* dengan perubahan parameter *minimum sample*



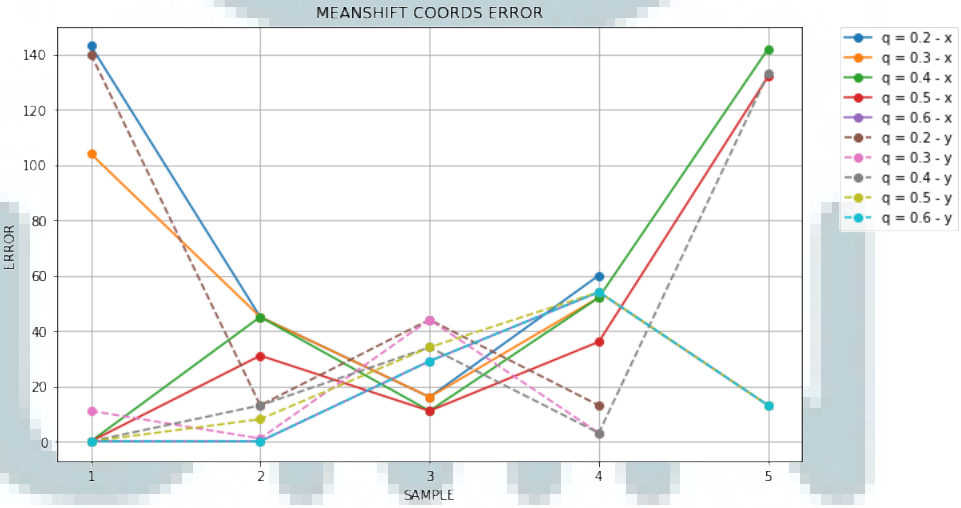
Gambar 5.13 *Distance error* berdasarkan *coordinates distance* pada *DBSCAN* dengan perubahan parameter *epsilon*

Pada algoritma *k-means* dan *meanshift*, hasil *distance error* titik *x* dan *y* dapat dilihat pada gambar 5.14 dan 5.15. *k-means* memiliki hasil yang sangat fluktuatif dimana pada setiap sample hasil *distance error* cukup tinggi dan tidak stabil dibandingkan dengan penggunaan *dbscan*. Pada sample ke-6 *k-means* gagal melakukan *clustering* pada percobaan saat merubah besarnya *k*. Hasil yang tidak

lebih baik dari uji coba menggunakan *meanshift* dimana seluruh nilai *quantile* tidak dapat melakukan *clustering* terhadap kumpulan *window* yang terdeteksi pada sample ke-6. Hal ini terjadi dikarenakan jumlah *window* yang terdeteksi sedikit dan sebaran masing-masing *data points* memiliki rentang yang pendek atau sempit.



Gambar 5.14 *Distance error* berdasarkan *coordinates distance pada k-means* dengan perubahan parameter *k*

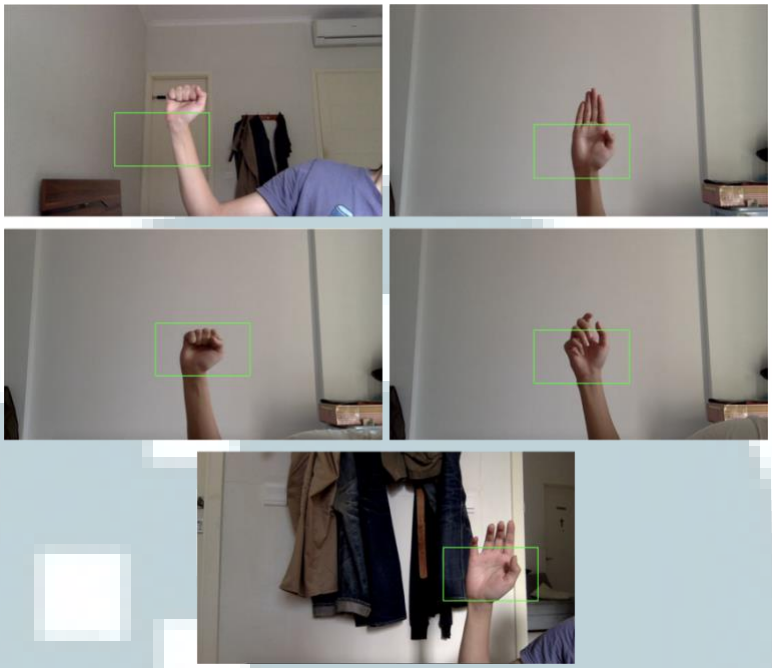


Gambar 5.15 *Distance error* berdasarkan *coordinates distance* pada *mean shift* dengan perubahan parameter *quantile*

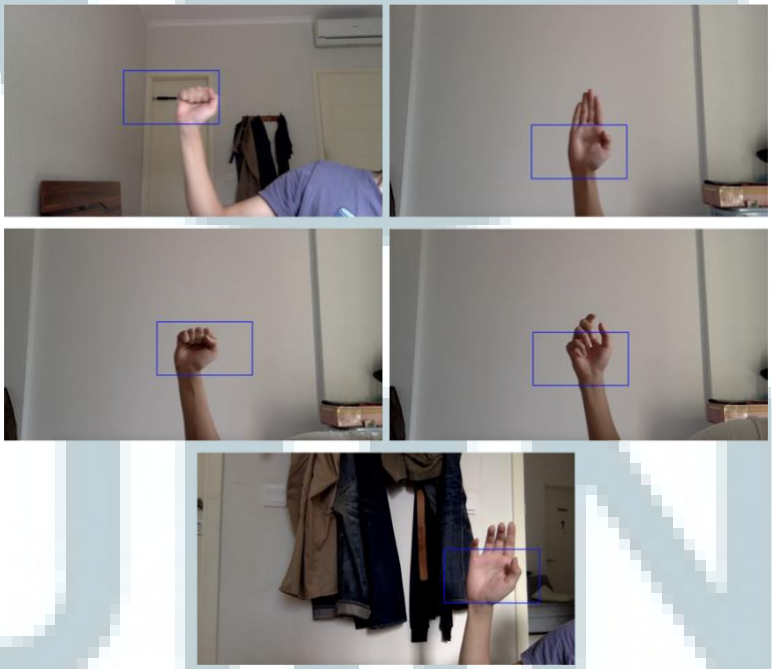
Dari hasil pengujian terhadap 3 algoritma *clustering* untuk membantu model pelacak tangan dalam melakukan lokalisasi, algoritma *dbscan* masih menjadi algoritma yang memberikan performa lebih baik dibandingkan *k-means* dan *meanshift*. Algoritma *dbscan* juga baik dalam melakukan *clustering* terhadap dataset yang jumlahnya cukup sedikit. Namun, jika menggunakan algoritma *k-means* atau *mean shift* pada jumlah *window* yang terdeteksi sedikit dan gagal untuk melakukan *clustering*, $1 < n < 4$ dimana jika kumpulan *window* yang terdeteksi lebih besar dari 1 dan lebih kecil dari 4 maka hasil lokalisasi yang digunakan adalah *centroid* yang diambil dari rata-rata *x* dan *y* seluruh *window* yang terdeteksi.

5.2.5 Hasil Lokalisasi Model Pelacak Tangan pada Video

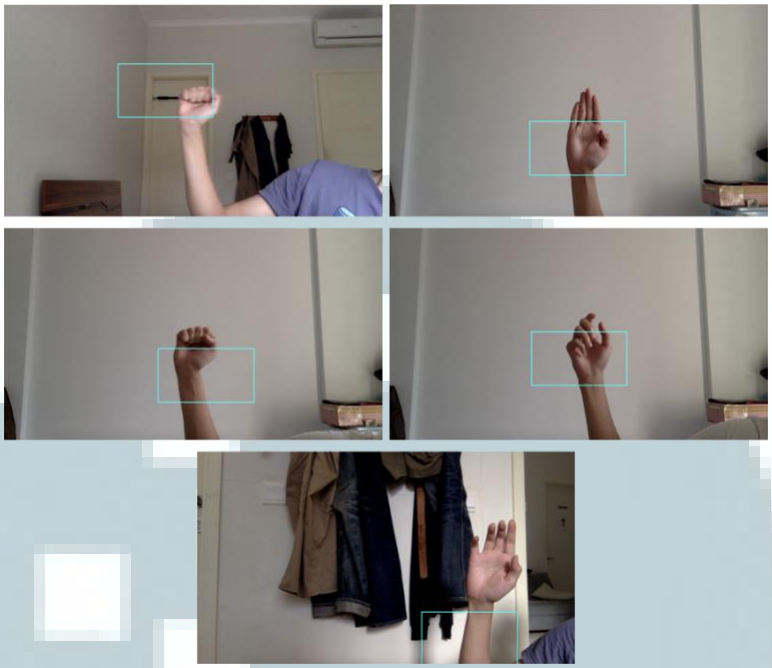
Lokalisasi yang dilakukan model pelacak tangan adalah berupa *bounding box* pada *region of interest* yang ditentukan dari rata-rata seluruh entitas dalam *cluster* modus. Lokalisasi dilakukan pada *frame* pertama pada sebuah video. Dari 5 sample yang telah direkam, pada bagian ini hasil lokalisasi adalah hasil lokalisasi menggunakan parameter terbaik pada 3 algoritma *clustering* yang digunakan. Gambar 5.16 merupakan hasil lokalisasi menggunakan *dbscan* dengan parameter *epsilon* = 0.2 dan *minimal sample* = *5*. Gambar 5.17 merupakan hasil lokalisasi menggunakan *dbscan* dengan parameter *epsilon* = 0.5 dan *minimum sample* = 4. Gambar 5.18 merupakan hasil lokalisasi menggunakan *k-means* dengan parameter *k* = 3. Gambar 5.19 merupakan hasil lokalisasi menggunakan *meanshift* dengan parameter *quantile* = 0.5.



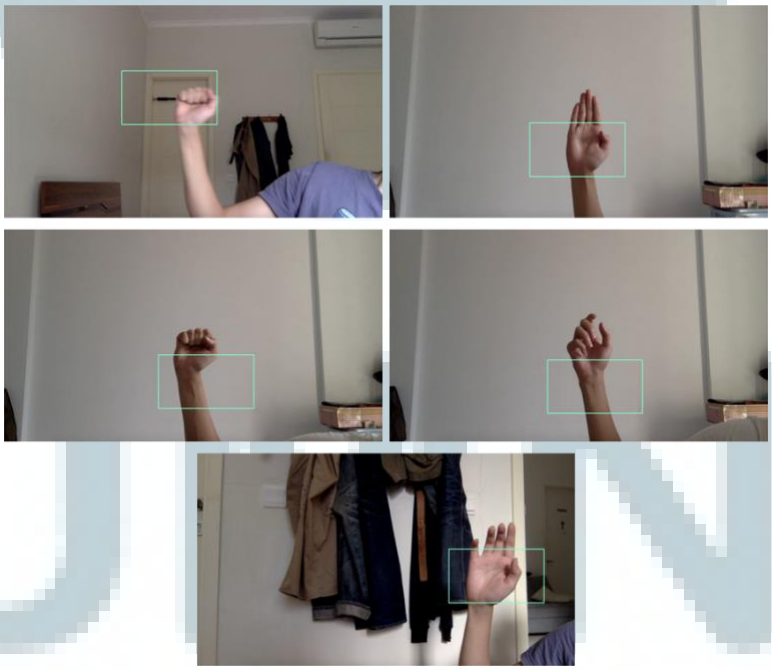
Gambar 5.16 Lokalisasi model pelacak tangan menggunakan *DBSCAN* dengan parameter *epsilon* = 0.2 dan *minimum sample* = 5



Gambar 5.17 Lokalisasi model pelacak tangan menggunakan *DBSCAN* dengan parameter *epsilon* = 0.5 dan *minimum sample* = 4



Gambar 5.18 Lokalisasi model pelacak tangan menggunakan *k-means* dengan parameter *k* = 3



Gambar 5.19 Lokalisasi model pelacak tangan menggunakan *mean shift* dengan parameter *quantile* = 0.5

Dari gambar 5.16, 5.17, 5.18 dan 5.19, hasil lokalisasi menggunakan *DBSCAN* pada gambar 5.16 dan 5.17 merupakan hasil yang paling presisi dibandingkan dengan hasil lokaliasi pada gambar lainnya yang menggunakan algoritma lain. Namun dengan menggunakan *dbscan* dengan parameter sesuai dengan gambar 5.16, hasil lokalisasi terbukti kurang presisi dimana *bounding box* pada sample ke-1 ada pada pergelangan tangan. Kemudian hasil lokalisasi menggunakan algoritma *k-means* terbukti kurang presisi pada sample ke -3 dan ke-5 begitu juga dengan penggunaan algoritma *mean shift* pada gambar 5.19, dimana pada sample ke-3 dan ke-4 hasil lokalisasi kurang presisi pada tangan.