Bonus project

Μαθητής: Κωνσταντίνος Δρακάκης

ΑΜ: TH20049

Εκφώνηση: Να κατασκευαστεί αλγόριθμος ο οποίος θα λαμβάνει εικόνα σε πραγματικό χρόνο από κάμερα (RGB) και θα εμφανίζει στην εικόνα της οθόνης το πλαίσιο συντεταγμένων του ανθρώπινου χεριού. Το πλαίσιο θα πρέπει να είναι τοποθετημένο στην αρχή του αντίχειρα με τον άξονα x να δείχνει προς τον δείκτη και τον άξονα z να είναι κάθετος με τον x και όσο πιο κοντά γίνεται στην κατεύθυνση του αντίχειρα.

Tip: Θα πρέπει να χρησιμοποιηθεί έτοιμη βιβλιοθήκη η οποία αναγνωρίζει κόμβους των δακτύλων του ανθρώπινου χεριού.

Αρχείο: MiniMachineVisionProject.py

Για να εντοπίσουμε άμεσα τον αριθμό των διαθέσιμων πηγών βίντεο (κάμερες), αξιοποιούμε τη συνάρτηση enumerate_cameras. Μέσω ενός γραφικού περιβάλλοντος (GUI), επιτρέπουμε στον χρήστη να επιλέξει την επιθυμητή κάμερα για χρήση.

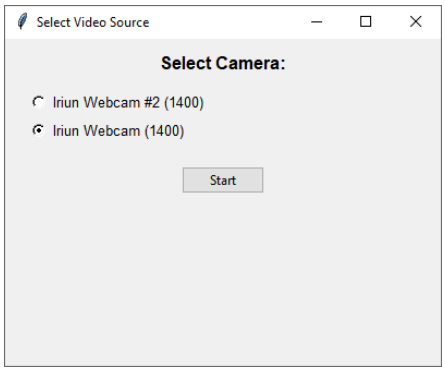


Figure 1

Ένα διάνυσμα στον τρισδιάστατο χώρο μπορεί να οριστεί ως η διαφορά δύο σημείων. Σκοπός της εργασίας είναι να προβάλουμε ένα σημείο από τον τρισδιάστατο χώρο πάνω σε μια δισδιάστατη επιφάνεια στην προκειμένη περίπτωση, την επιφάνεια καταγραφής της κάμερας.

Αρχικά, υποθέτουμε ότι υπάρχει ένα αδρανειακό σύστημα συντεταγμένων και ένα σημείο στο χώρο. Μια κάμερα, που ακολουθεί το κλασικό μοντέλο pinhole, είναι στραμμένη προς αυτό το σημείο. Σύμφωνα με το μοντέλο αυτό, θεωρείται ότι το φως περνά από μια ιδανικά μικρή οπή και προβάλλεται σε ένα επίπεδο (εικονικό επίπεδο προβολής). Επίσης, ο φακός της κάμερας προκαλεί ραδιοειδείς και εγκάρσιες παραμορφώσεις, που πρέπει να διορθωθούν.

Ο στόχος μας είναι να μετατρέψουμε τις συντεταγμένες του σημείου από το παγκόσμιο σύστημα αναφοράς στο σύστημα της κάμερας μέσω ενός ομογενούς μετασχηματισμού. Στη συνέχεια, να εφαρμοστεί προοπτική προβολή (perspective projection), δηλαδή η διαίρεση των οριζόντιων και κάθετων συντεταγμένων με τη συντεταγμένη βάθους Z, με αποτέλεσμα τη μετάβαση στο εικονικό επίπεδο προβολής (ένα νοητό επίπεδο μπροστά από την κάμερα).

Το επόμενο βήμα είναι η μετατροπή του σημείου αυτού από το εικονικό επίπεδο σε pixels στο επίπεδο της εικόνας, με τη χρήση του intrinsic camera matrix. Αυτή η διαδικασία απαιτεί την ευθυγράμμιση του οπτικού κέντρου (optical center), δηλαδή το κέντρο του φακού, τόσο με το εικονικό επίπεδο όσο και με το πραγματικό επίπεδο της εικόνας.

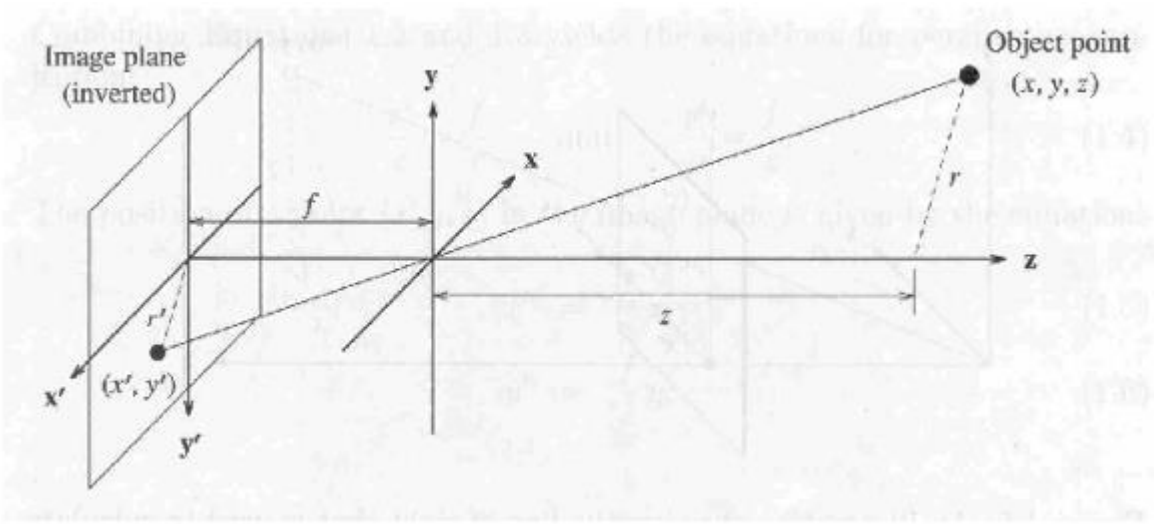


Figure 2

Επιπλέον το σύστημα συντεταγμένων του χώρου σύμφωνα με αρκετά διαδεδομένες συμβάσεις είναι XYZ ενώ των καμερών είναι ZXY (το z προς τα έξω).

Τέλος, στη συγκεκριμένη περίπτωση, το σημείο που επεξεργαζόμαστε είναι ήδη εκφρασμένο στο σύστημα συντεταγμένων της κάμερας, καθώς έχουμε άμεση πρόσβαση στις τρισδιάστατες συντεταγμένες του μέσω ενός τεχνητού νευρωνικού δικτύου, εκπαιδευμένου σε μεγάλο αριθμό εικόνων. Το δίκτυο αυτό είναι ικανό να εκτιμά το βάθος και τις σχετικές θέσεις των σημείων ως ποσοστά ή απόλυτες μονάδες μέσα στο οπτικό πεδίο της κάμερας.

Επομένως ένα τεχνητό νευρωνικο δίκτυο εκτιμά ως ποσοστά % ποσό πάνω κάτω δεξιά η αριστερά σε σχέση με το εικονικό επίπεδο προβολής είναι το σημείο για αυτό και εξίσου σημαντικό είναι το κέντρο του εικονικού επίπεδου προβολής και της κάμερας (optical center).

Η βιβλιοθήκη MediaPipe δεν εκτιμά αποστάσεις σε μέτρα. Η τιμή z είναι σχετική, τυπικά μεταξύ ~0 και -1, και πρέπει να μετατρέπεται εμπειρικά για αυτό το λόγο την μετατρέπουμε σε μια τιμή που ανήκει σε ένα εύρος στο πρόγραμμα από 150 mm έως 1000 mm.

Διαφορετικά θα χρειαζόμασταν η δυο κάμερες που στην ουσία από κάθε κάμερα θα "εκτοξευόταν" μια ακτίνα που περνά από αυτό το σημείο και εκεί που οι δύο ακτίνες τέμνονται στον χώρο θα βρισκόταν το σημείο.

Για να συνοψίσουμε, μεταβαίνουμε στο δυσδιάστατο χώρο με τους εξής μετασχηματισμούς:

[Σύστημα συντεταγμένων κόσμου] -ομογενής μετασχηματισμός-> [Σύστημα σ. κάμερας]

-perspective projection-> [εικονικό επίπεδο προβολής (ένα επίπεδο μπροστά από κάμερα) *1]

-Camera matrix-> [pixels στο επίπεδο της εικόνας *2]

*Για αυτό τον λόγο χρειάζεται το κοινό και πραγματικό optical center μεταξύ *1&*2

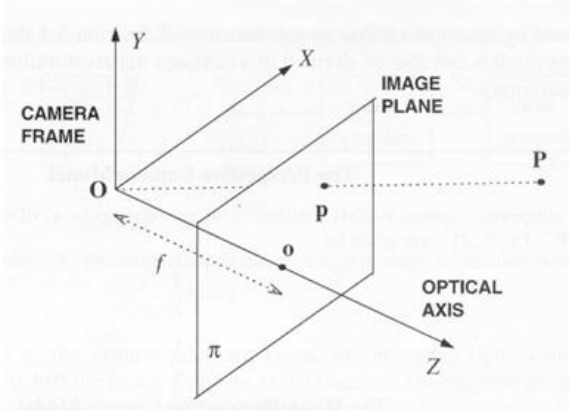


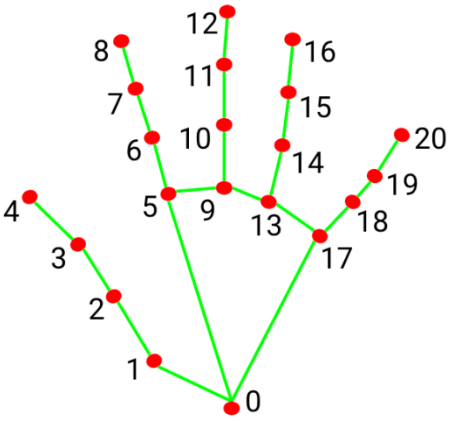
Figure 3

Αφού εξηγήσαμε τα παραπάνω στη συνάρτηση project_point_3d_to_2d(point_3d)

- Ορίζουμε τον μετασχηματισμό camera_matrix που “μας πηγαίνει από το εικονικό επίπεδο προβολής σε pixels στο επίπεδο της εικόνας”.
- Παράμετροι, ροδοειδής και εγκάρσιας παραμόρφωσης
- Rvec = tvec = 0 διότι είμαστε ήδη στο σσ της κάμερας

Η συνάρτηση cv2.projectPoints(points_3d, rvec, tvec, camera_matrix, dist_coeffs) έχει υπόψη της όλα όσα έχουν αναφερθεί.

Αφού έχουμε ανιχνεύσει το δεξί μας χέρι με την βιβλιοθήκη mediapipe περνούμε τα σημεία ενδιαφέροντος που είναι 3 , index_mcp, index_tip και thumb_tip.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 4

Αξιοποιώντας αυτά τα 3 σημεία κατασκευάζουμε 2 διανύσματα που είναι κάθετα μεταξύ τους συν ένα ακόμα που είναι κάθετο στο επίπεδο που σχηματίζουν τα προηγούμενα δυο και στη συνέχεια ακολουθεί κανονικοποίηση και κλιμάκωση.

Η γραμμική αλγεβρα πιθανός να ανήκει στα αγαπημένα μαθηματικά του αναγνώστη όμως ένας πολύ ωραίος τρόπος περιγραφής της κατασκευής τριών κάθετων διανυσμάτων στο τρισδιάστατο χώρο περιγράφεται στο επόμενο βίντεο Construction process of 3D frame με το manim ένα framework που δημιουργήθηκε από τον άνθρωπο πίσω από το κανάλι 3blue1brown.

Τέλος είναι η απαραίτητη η βελτίωση του μοντέλου με καθορισμό των απαραίτητων παραμέτρων που βρίσκονται από το πρόγραμμα calibration.py με μια σκακιέρα

Camera Calibration Pattern Generator – calib.io

Αποτέλεσμα:

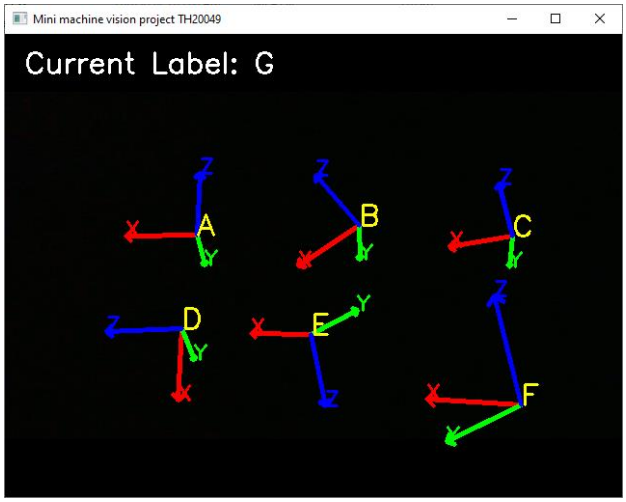


Figure 5

(Είναι μαύρο διότι αργότερα κλείσαμε την κάμερα) (webcam connected to pc)

Παρατηρήστε ότι όσο απομακρυνόμαστε από τον φακό «τα πάντα μικραίνουν» (Frame C) και αντίστροφα μεγαλώνουν (Frame F) μια από τις ιδιότητες του μοντέλου (figure 6)

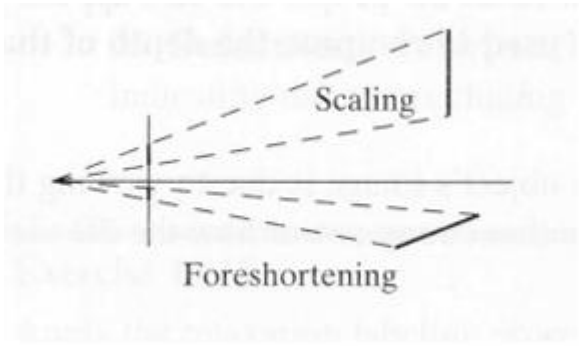


Figure 6

Για κάθε πινακίδα (label) έχουμε έναν μετρητή που προσπελάζει κάθε κεφάλαιο γράμμα στον πίνακα ascii.

Link to GitHub repo:

kostasCode/Robotics-I-8.012 -> BonusPoseDetectionCameraCalibration