Vaja 4: Računalniški vid v robotiki

Robotika in računalniško zaznavanje

2017/2018

Za vajo ustvarite mapo `vaja4`. Sem si prenesite tudi razpakirano vsebino datoteke `vaja4.zip` s spletne strani predmeta. Rešitve nalog boste pisali v *Matlab/Octave* skripte in jih shranili v mapo `vaja4`. Da uspešno opravite vajo, jo morate predstavili asistentu na vajah. Pri nekaterih nalogah so vprašanja, ki zahtevajo skiciranje, ročno računanje in razmislek. Odgovore na ta vprašanja si zabeležite v pisni obliki in jih prinesite na zagovor. Deli nalog, ki so označeni s simbolom ★ niso obvezni. Brez njih lahko za celotno vajo dobite največ 75 točk (zgornja meja je 100 točk kar pomeni oceno 10). Vsaka dodatna naloga ima zraven napisano tudi število točk. V nekaterih vajah je dodatnih nalog več in vam ni potrebno opraviti vseh.

Uvod

Zaključna vaja povezuje robotiko in računalniško zaznavanje. Vaš robotski sistem boste nadgradili s kamero, ki bo opazovala delovno površino robotskega manipulatorja, preko zajete slike pa boste lahko na površini zaznali objekte glede na njihovo barvo in obliko. Preko homografske preslikave boste točke iz koordinatnega prostora slike pretvorili v prostor robota ter mu ukazali, naj se premakne tako, da pokaže na ustrezni predmet.

Naloga 1: Homografska transformacija

Homografska transformacija je pogosto uporabljana v projektivni geometriji. Predstavlja projekcijo ene ravnine v prostoru na drugo ravnino v prostoru. Naj bo $\mathbf{x}_w$ točka na ravnini v svetovnih koordinatah in naj bo $\mathbf{x}_c$ točka v koordinatah slikovne ravnine. Natančneje, $\mathbf{x}_w = [x_w, y_w, 1]^T$ in $\mathbf{x}_c = [x_c, y_c, 1]^T$. Projekcijo točke v svetovnih koordinatah $\mathbf{x}_w$ v točko $\mathbf{x}_c$, ki se nahaja na slikovni ravnini senzorja, preslikamo s homografijo $\mathbf{H}_w^c$ po enačbi

$$\mathbf{x}_c = \lambda \mathbf{H}_w^c \mathbf{x}_w, \tag{1}$$

kjer je homografija matrika velikosti 3×3 elementov

$$\mathbf{H}_w^c = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}. \tag{2}$$

Relacija v enačbi (1) je zapisana v homogenih koordinatah in je zato napisana le do skale natančno. V praksi ima matrika $\mathbf{H}_w^c$ samo 8 *prostih* parametrov in ne 9, saj zadnji element postavimo na ena, $h_{33} = 1$. Če hočemo točke v koordinatah slike $\mathbf{x}_c$ preslikati v točko $\mathbf{x}_c$, ki se nahaja na svetovnih koordinatah, moramo uporabiti inverz homografije $\mathbf{H}_c^w = (\mathbf{H}_w^c)^{-1}$.

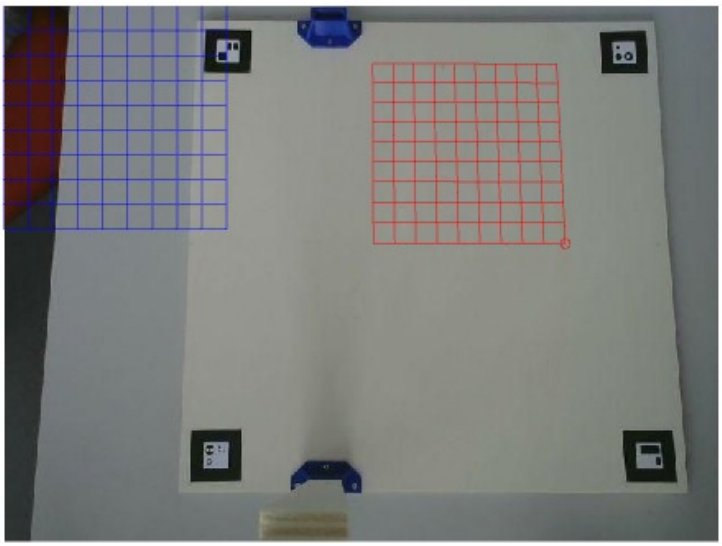
V naslednjih nalogah boste preizkusili homografsko transformacijo na pripravljenih podatkih.

(a) Naložite sliko iz datoteke `camera1.jpg` ter homografsko matriko iz datoteke `camera1.txt` z uporabo funkcije `load`. Prikažite sliko ter na njej izrišite mrežo točk s spodnjim primerom.

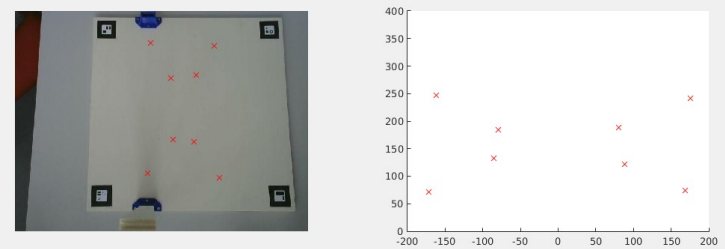
```
n = 10;
X = linspace(0, 200, n);
Y = linspace(0, 200, n);
figure(1); imshow(I); % Draw the image
hold on;
for i = 1:n
    plot([X(i), X(i)], [Y(1), Y(end)], 'b');
    plot([X(1), X(end)], [Y(i), Y(i)], 'b');
end
hold off;
```

(b) Izris spremenite tako, da točke črt pred izrisom preslikate s prebrano homografsko matriko. Pri tem morate točke najprej preslikati v homografske koordinate (na konec vektorja dodati vrednost 1), jo pomnožiti z matriko, nato pa *točko pretvoriti nazaj iz homogenih koordinat* tako, da vektor delimo z vrednostjo zadnjega elementa ter tega nato odstranimo.

$$p_S = [x, y, 1], p_H = H * p_S = [x_H, y_H, z_H], p_D = [x_H/z_H, y_H/z_H]$$



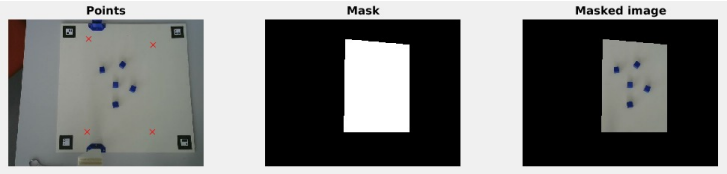
(c) V okviru te naloge si bomo ogledali še preslikavo v drugo smer. To tehniko lahko uporabimo za preslikavo iz slikovnih koordinat v svetovne koordinate. Matriko `H` invertirajte z uporabo funkcije `inv`. Prikažite sliko ter z uporabo funkcije `ginput` vnesite eno ali več točk s klikom na sliko. Dane točke preslikajte z uporabo inverza matrike H iz ravnine kamere v svetovne koordinate ter jih prikažite v novem grafu z uporabo funkcije plot.



Naloga 2: Detekcija objektov

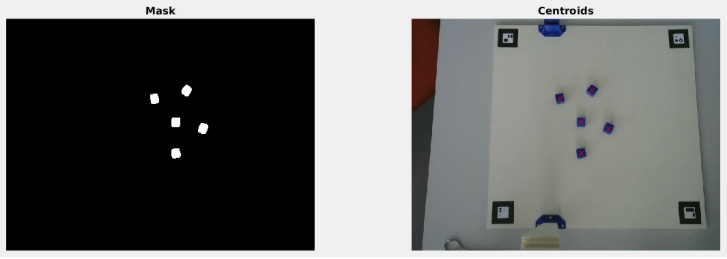
V tej nalogi boste implementirali preprosto detekcijo planarnih objektov z uporabo segmentacije, morfoloških operacij ter opisa regij. Sledite naslednjim korakom (lahko si pomagate z detekcijo bombonov, če ste jo implementirali v okviru prve vaje).

(a) Naložite sliko iz datoteke `camera2.jpg` ter jo prikažite na zaslonu. Z uporabo funkcije `ginput` določite meje poligona delovne površine kot je prikazano na spodnjem primeru. Nato za podane točke generirajte rastersko masko z uporabo funkcije `poly2mask`, ki na vhodu sprejme koordinate poligona ter velikost rasteriziranega območja (za lažjo uporabo naj bo ta enaka velikosti slike). Če ste pri prvi vaji implementirali funkcijo `immask`, jo sedaj uporabite za prikaz izolirane delovne površine.



(b) Sliko pogleda kamere pretvorite v HSV barvni prostor ter z uporabo upragovanja določite regije zelene barve. Po potrebi segmentacijo izboljšajte z uporabo morfoloških operacij `erode` in `dilate`, da se znebite šuma. Končno binarno masko združite z masko delovne površine, da odstranite morebitne napačne detekcije.

(c) Uporabite funkcijo `bwlabel` da masko razdelite na posamezne komponente. Za vsako komponento centroid (npr. z uporabo funkcije `moment` iz prejšnje vaje). Centroide prikažite kot točke na izvorni sliki iz datoteke `camera2.jpg`.



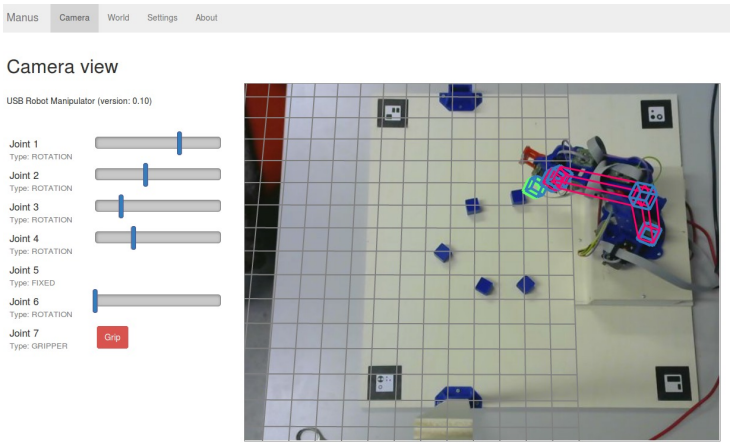
Naloga 3: Krmiljenje robotskega manipulatorja

V zadnji nalogi povežite prvi dve nalogi ter prejšnje vaje ter implementirajte naslednje korake (vsi skupaj tvorijo eno samo *Matlab/Octave* skripto). Za to nalogo morate imeti

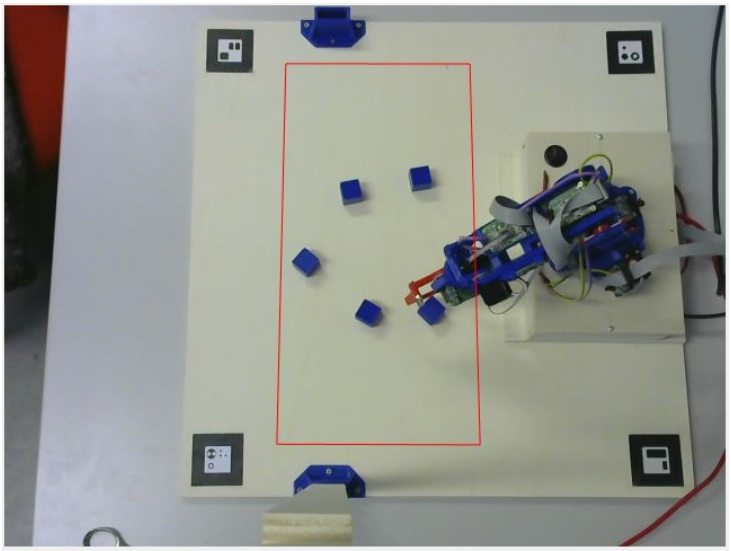
vzpostavljno okolje za uporabo manipulatorja (navidezni stroj ali dostop do pravega manipulatorja v laboratoriju).

V primeru, da boste delali doma, poženite simulator v načinu s statično sliko, ali pa natisnete A4 kalibracijsko podlago, s pomočjo katere sistem ustrezno določi homografsko preslikavo. Le-to imate na voljo v datoteki `surface.pdf` v materialu vaje. Ko bo sistem zaznal kalibracijsko podlago, bo v sliko pravilno umeščen tudi robotski manipulator. V primeru, da boste hoteli v navidezni stroj vključiti spletno kamero, boste morali v VirtualBox namestiti razširitve[1], potem pa kamero v stroj povezati preko menija `Devices > Webcams`.

V primeru, da boste pri testeranju uporabili navidezni manipulator (rezultate si v tem primeru poglejte v spletnem vmesniku), na koncu v vsakem primeru delovanje preverite tudi na pravem robotskem manipulatorju v fakultetnih prostorih.



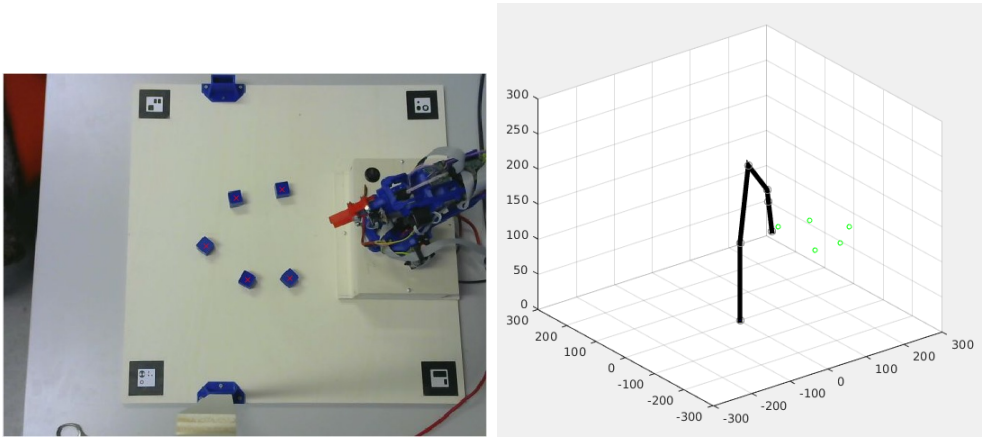
(a) Postavite sistem za dostop do manipulatorja in kamere ter si poglejte primer `demo_camera`, ki iz kamere pridobi sliko ter jo prikaže na zaslonu. Demonstracijski program prebere tudi homografsko matriko, vi pa razširite skripto tako, da bo na sliko izrisala kvadrat z (x, y) koordinatami $(100, -200)$, $(100, 200)$, $(300, 200)$, $(300, -200)$ v svetovnem koordinatnem sistemu delovne ravnine.



[1]VirtualBox Extension Pack ki ga za svojo verzijo orodja VirtualBox najdete na `http://download.virtualbox.org/virtualbox/`.

(b) Skripto razširite tako, da bo na sliki zaznala modre kocke ter na prikažite njihove centroide. Če delate doma in kock nimate na voljo, si lahko pomagate tudi s papirnatimi liki, ki jih imate na voljo v datoteki `objects.pdf` v okviru materiala vaje (v tej nalogi uporabite zgolj kvadrate različnih barv). Po potrebi lahko določite tudi masko delovne površine (ker je kamera v scenariju statična, lahko to masko določite enkrat ter jo potem uporabljate večkrat).

(c) Preslikajte vse točke iz koordinatnega sistema kamere v koordinatni sistem delovne površine. V zanki potujte preko seznama točk ter manipulator premikajte tako, da bo s koncem prijemala pokazal na posamezni zaznani objekt na delovni površini. Ker v $3D$ prostoru robotskega manipulatorja zaznane točke ležijo na delovni površini (koordinata v smeri z je 0) kot take niso neposredno primerne za podajanje cilja robotskemu manipulatorju. Zato točko preoblikujte tako, da ji podate novo višino od delovne površine (npr. $z = 30$). Take točke bodo od ravnine oddaljene 3cm.



(d) Razširite detekcijo objektov tako, da bo zaznala kocke dveh barv, modre in črne. Manipulator naj nato pokaže vse zaznane modre kocke z zaprtim prijemalom, nato pa še črne kocke z odprtim prijemalom.

(e) Implementirajte lastno idejo z uporabo analize kock na delovni površini in reakcijo manipulatorja na stanje (kock vam v obveznem obsegu naloge ni potrebno premikati, če pa jih boste, se to šteje kot dodatna naloga).

(f) ★ (15 točk) Napišite algoritem, ki zazna modre kocke ter jih z manipulatorjem eno za drugo premakne na levo stran delovne površine (če jih zazna na desni strani). Bodite pozorni na robustnost prijemanja ter na to, da prijemala ne poškodujete (ne rinite ga v delovno površino, kock ne stiskajte preveč). Naloga je dokaj odprta, saj morate sami implementirati približevanje manipulatorja kocki, da bo prijemalo pod pravim kotom. Seveda pa si lahko pomagate s kodo, ki je priložena v okviru materiala za vaje.

(g) ★ (15 točk) Implementirajte lastno idejo uporabe robotskega manipulatorja, ki presega zgolj prepoznavanje kock brez manipulacije. Ideja lahko razširi razpoznavanje (objekte, ki niso kocke) ali manipulacijo (torej premikanje objektov). Število točk je odvisno od zahetevnosti ideje in njene realizacije.