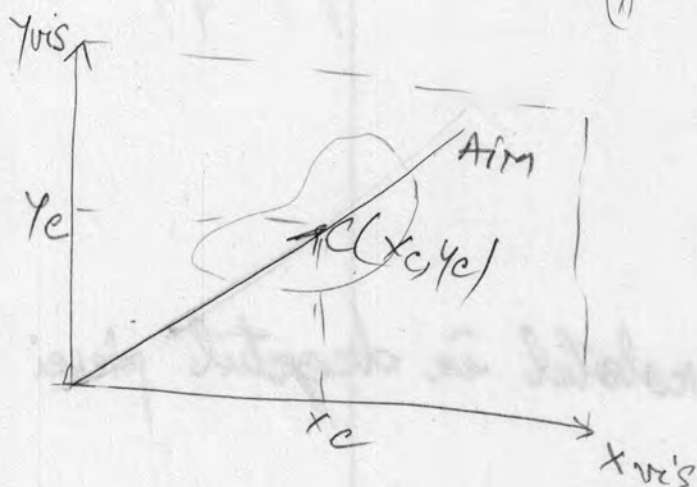
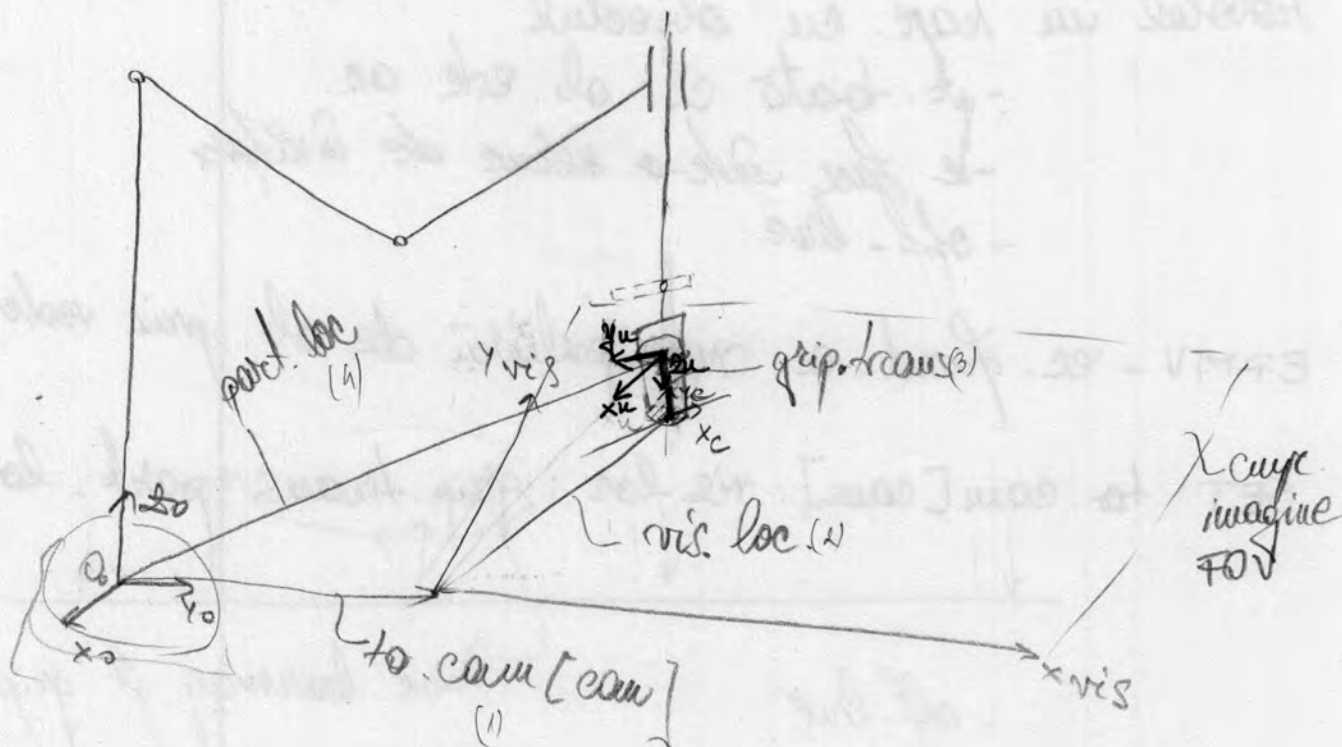


Legătura vedere-mișcare



Aim - axe de direcție mișc

x_c, y_c - coord. centrului de greutate

CV - cam. video

SET part. loc = to.cam[cam] : vis.loc : grip.trans (EFM) → în timp real.

- (1) to.cam[cam] - ff. ^{off-line} camera-robot ^{cam. prog} ^{off-line} grip.trans (EFM)
- se calc. o sq. dată, indiferent de apl. cu ^{relativă a} cam. cu robot.
 - se obt. pentru o sesiune de calibrare a CV.
- (2) vis.loc - indică ^{locatie} poz. și orientarea ob. în pl. imagine
- se obt. pl. fiecare ob. ce treb. accesat de robot
 - VLOCATE

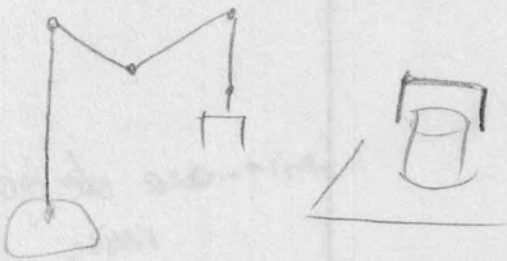
- (3). grip.trans - este o #. care arată cum reb. să fie poz. robotul în rap. cu obiectul
- pt. bato cl. ob. este ac.
 - se face încă o estimare de înălțare,
 - off-line

EFMV - ec. fund. a manipulării de ob. prin vedere

SET to.cam[cam]: vis.loc : grip.trans = part.loc.

-off-line

→ trinc. learning pt. grip.trans



HERE part.loc ; aduc robotul în dreptul pteiei

SCR

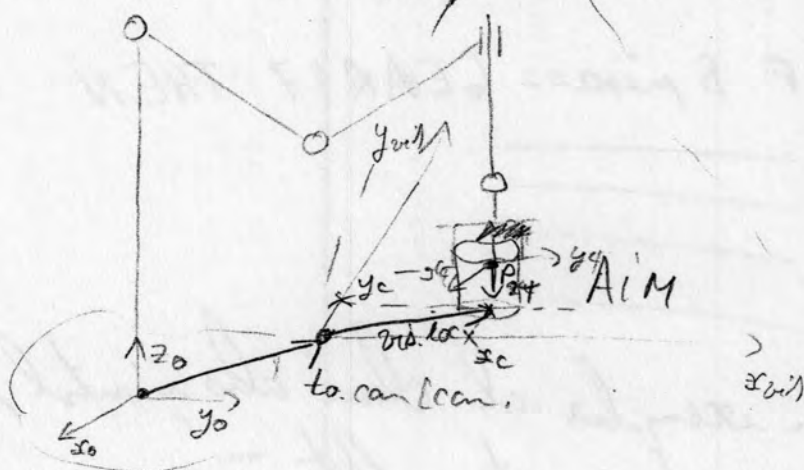
curs(S8)

5.04.2012

Ec. fund. a mișc. (EFM) → ss. de calib. camera-relect
VLOCATE → t-r pt fix. în obiect
SET part. loc = to. cam (can): vis. loc. grip. trans în reg. prie

Funcționare de timp-real

MOVE (s) part. loc



cam = camera virtuală
↳ ut de person
cu aj. camera
interpusă în
în funcție de apl.
dorită

AIM = axa de învârtire minimă

grip-trans → calc. off-line

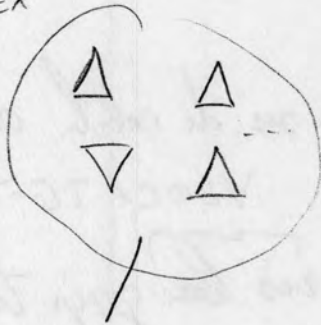
→ transf. de prindere pt. toate obiectele dintr-o clasă

→ calc. se face în subrutina de comandă a mișcării

CS/SOR

În VA pt. reb. (recunos. formelor - pattern recognition)
↳ vorbim de ob. identice ca aspect

Ex



clasă ← nume

↓
Sir de caract ASCII

Ex: "GEAR47"

\$piese

IF \$piese == GEAR47 THEN

Prototip, model - un exemplar al clasei ale cărei obiecte
a învățat sist. robot să recunoască
modelul de recunoaștere și modelul de
localizare

VTRAIN - multistructurată

↳ învățarea formei, aspectului
↳ -||- localizarea lin.

CS / SCR

VTRAIN. FINDER $\rightarrow$

VPLAN. FINDER $\rightarrow$

Ex: $\hat{T}$ Intr-o scenă am 3 cat de ob.

VTOUCHING

VOVERLAPPING

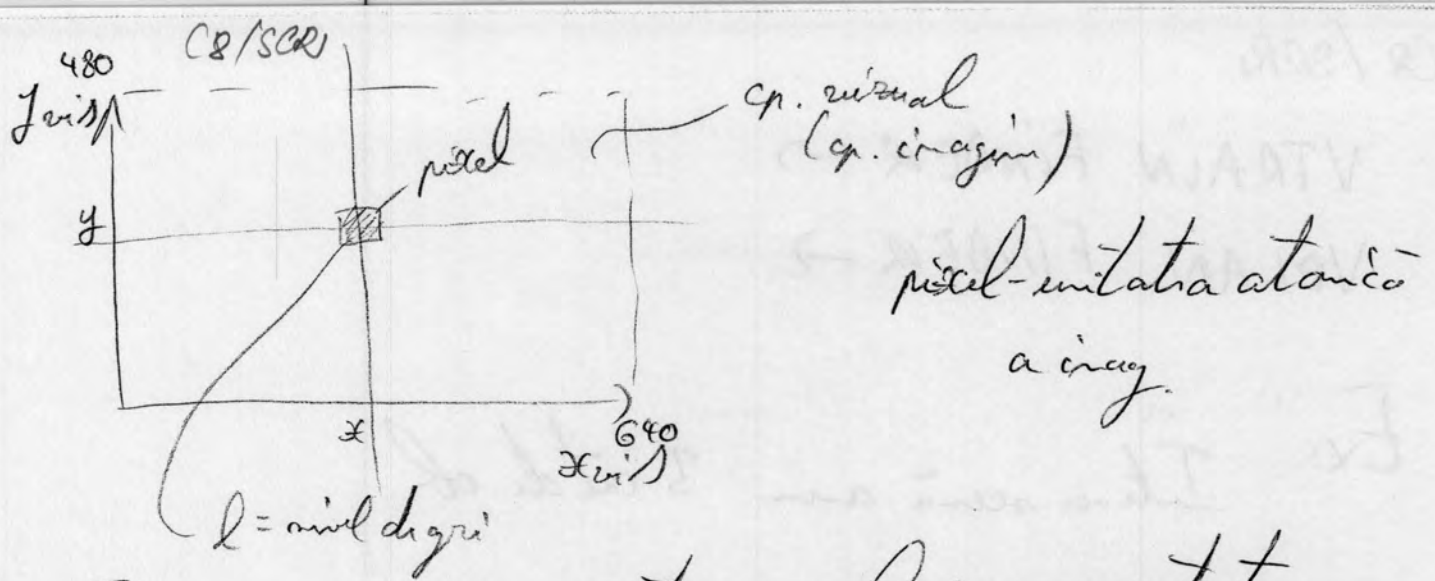
VFINDER - instruct de rec. direct

Format de raport. a imag.

- Imagine analogică $f(x, y) > 0$ x, y coord. p. ale
unui pct din imag
 $x, y \in \mathbb{R}^+$
- $\downarrow$
- Imag. numerică $f^*(x, y) \in \mathbb{Z}^+$

Această imag. numerică se obține prin

- esantionare p.
- cuantificarea a imag. analogice



$l = \text{nivel de gri}$
Formarea imaginii: $\rightarrow$ captura a luminii proiectate

$$f^* \rightarrow 256 \text{ de nivel de gri}$$

$$8 \text{ bits}$$

$$l \in [0, 255]$$

$$\uparrow \quad \uparrow$$

$$\text{min} \quad \text{max}$$

Imagie binarizata

$$\rightarrow f^*(i,j) = \begin{bmatrix} f_{i1} & \dots & f_{in} \\ \vdots & & \vdots \\ f_{m1} & \dots & f_{mn} \end{bmatrix} \quad f^*(i,j) = \begin{cases} 0, & f^*(i,j) \leq \text{thr} \\ 1, & f^*(i,j) > \text{thr} \end{cases}$$

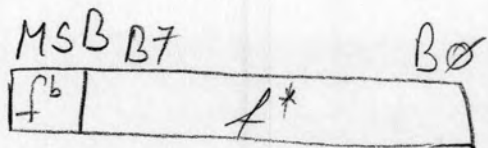
$$\text{thr} = \text{prag. de binarizare}$$

$$\text{thr} \in (0, 255)$$

V. THRESHOLD [can] $\rightarrow$ implicit 63

V. 2ND. THRESHOLD [can] $\rightarrow$ implicit 0

CR/SCR



$$m = \min(V. THRESHOLD, V. 2ND. THRESHOLD)$$

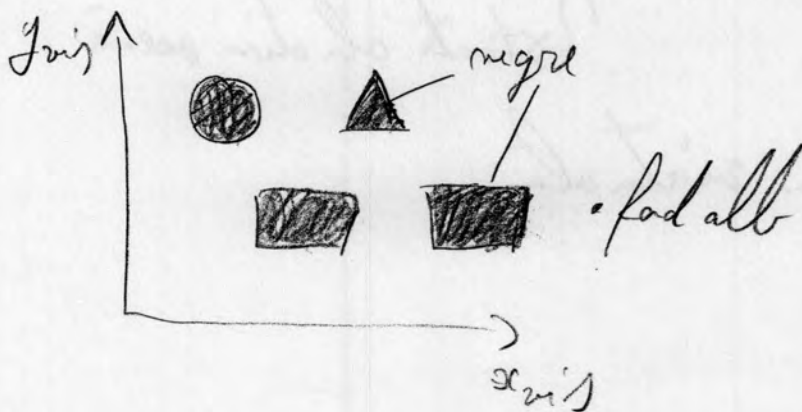
$$M = \max(\text{---} // \text{---})$$

$$f^b(x, y) = \begin{cases} 0, & f^* \in [m, M] \\ 1, & f^* \in [0, m] \cup [M, 255] \end{cases}$$

→ 80% glic de Vision apr. cu imag. binare

imagin' binaritate

VTRAIN.FINDER → multe instruct care apr
cu imag. de gr
(ObjectFinder → principiu)



V. BACKLIGHT

↳ switch

{ 1 fod alb
0 fod negru

ENABLE } mme-switch
DISABLE

Prel. de imagini

1. Low level image processing

2. H-level image processing

Obs: diferența este separarea directiilor față de fund

Separare

→ Se creează o BD pt fiecare direct

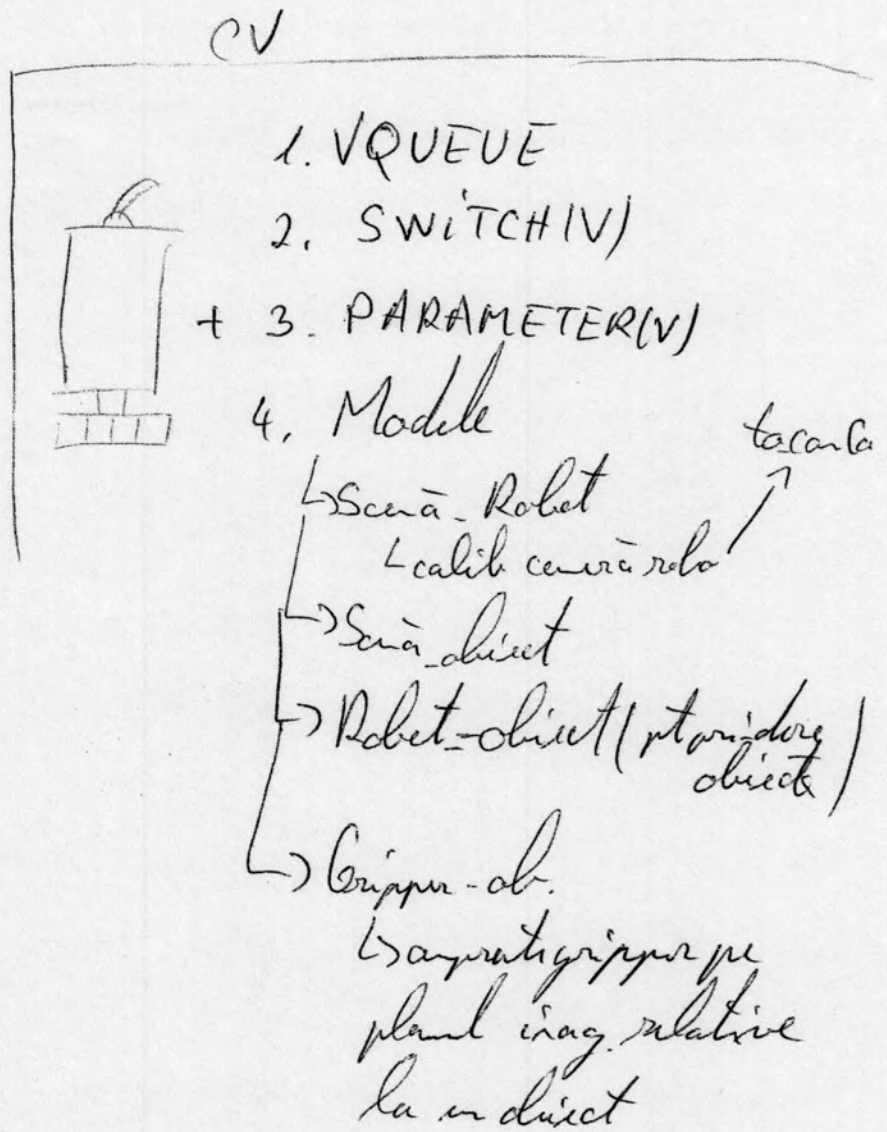
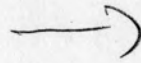
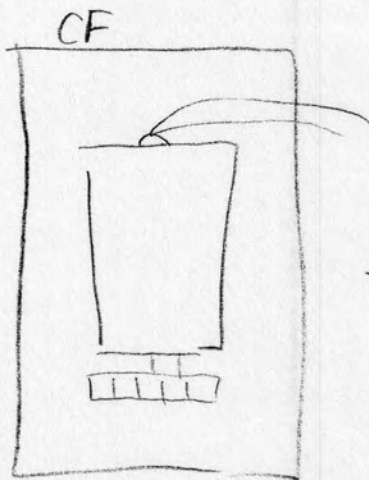
- info. contur
- aria obiect (pixels)
- nr de găuri
- lungimea conturului etc
- centru de masă (x_c, y_c)
- Aim (4 de orientare)

→ se află VQUEUE

→ toate ob. din scenă

VQUEUE e Cam. virtuală

CB/SCR



SCR

curs (Sg)

12.04.2012

miercuri
20
vineri

23-25 mai

Hilton

INCOM - Inform. Central
in Manufacturing

6-7 studenti

↳ 350 participati

WWW.incom.ro

↳ 6 celebritati

Andrew Curiale

TC 5.1 → Paul Valchenaers

↳ belgian

PROSA - Product Resource

Order Staff Agent

Multi-Agent Systems (MAS)

Product-Driven automation

09/10/20

→ Intelligent Products

↳ Duncan McFarlane
Kary Framery

→ abordare Holonică

Manufacturing
Execution
System
↓

→ 22 mai → workshop de 12c

↳ analiză, proiectare MES

↓
Open-source

Camera virtuală

în mod automat

→ set de date asociate unei camere fizice (sur. optice)
imagini achiziționate

↳ parametri

→ switch-uri (flag-uri)

→ modele - de calibrare (Camera) (S-R)

unic

- de recunoaștere (S-O)

- de prindere (R-O)

- de emitere celule (Gripper - Obiect)

pt. fixare
obiect
(localizare)

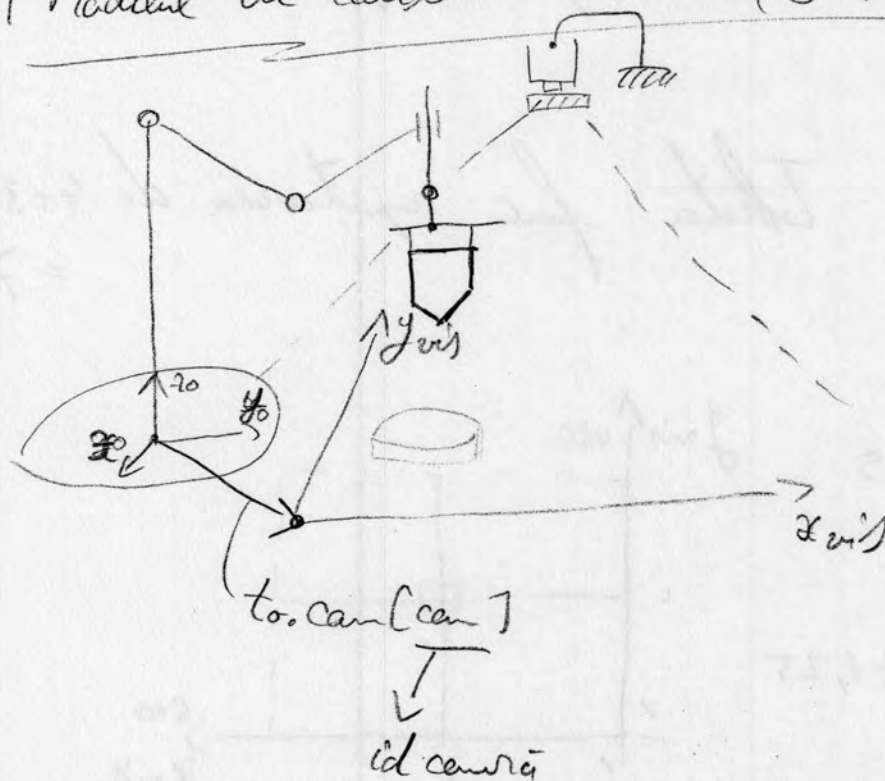
parametri

Off-line

de învățare

base de date

Modulul de calibrare C-R(S-R)

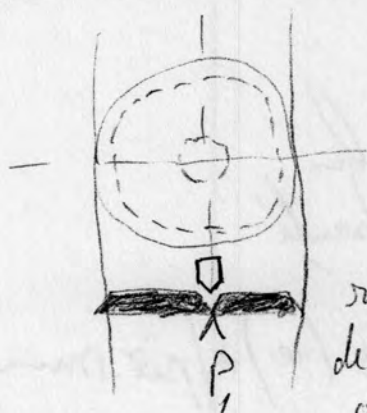


$to.cam[] \rightarrow TRANS(X, Y, Z, RX, RY, RZ)$

$$X, Y, Z = Pr \begin{vmatrix} (x_{vis}, y_{vis}, z_{vis}) \\ x_0, y_0, z_0 \end{vmatrix}$$

Se utilizează un disc circular

$$C(x_c, y_c) + AIM$$



scut
de m.m.
ori

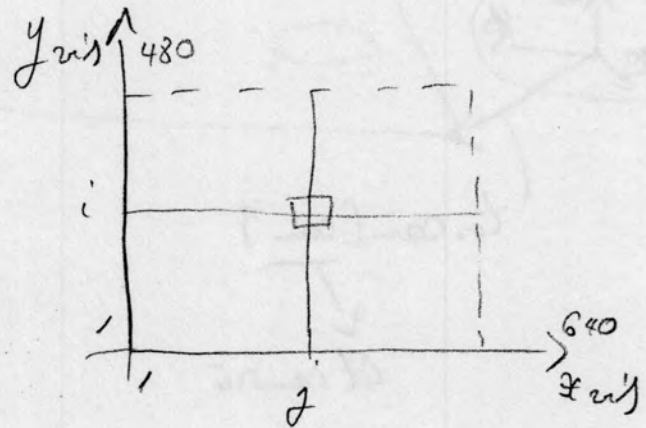
HERE $p_1 \rightarrow$ ref. robot
DEPARTS 100
MOVE safe
VPICTURE (cam, mode, st1,
st2) how many
ref. vision

Obs: robotul SCARA are coord. "Theta"

03/500
 Consider planul cu 4 cadre, în repet procedurape toate
 cadrele

- la reb cu capet Teheta fac repetarea de $4+3 \times 2 = 7$

→ pix. la. mm - 0,295
 XY scale } $0,75 \div 1,25$



Aria obiectelor

→ VFEBTURE(index)
 ↓
 ∈ 1,50

$f^*(i,j)$

→ VFEBTURE(10) ≡ aria obiectului
 în pixeli bruti

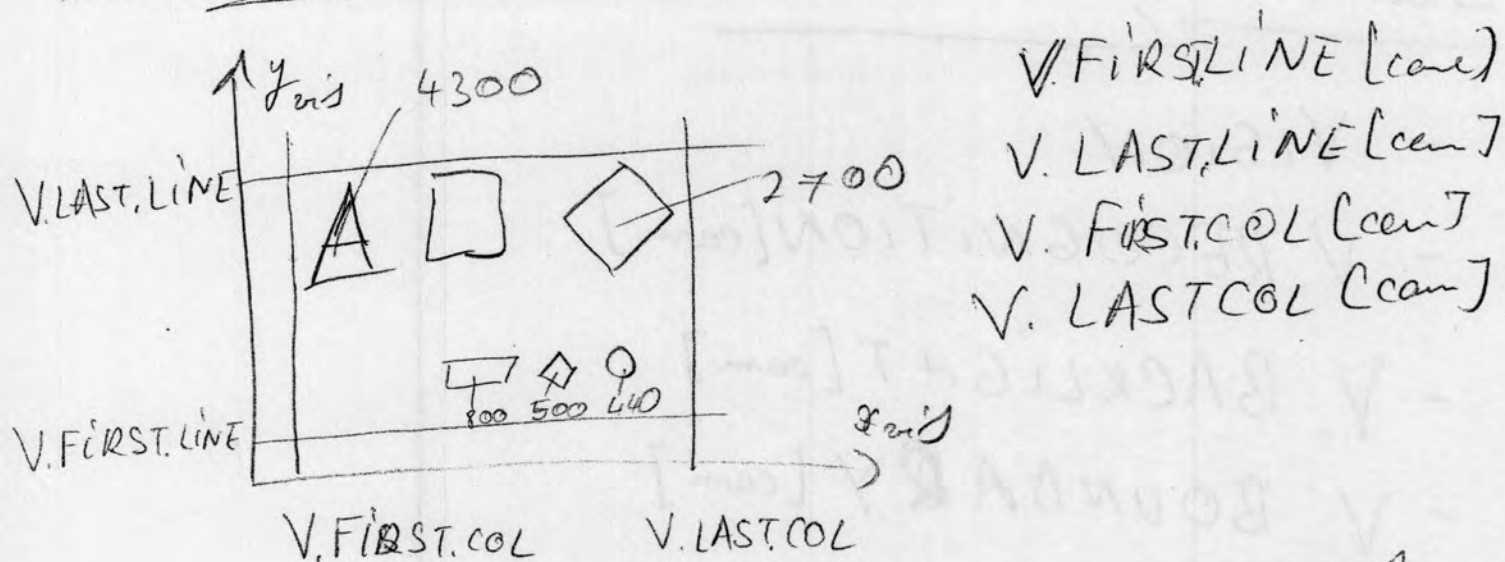
$$\text{ar. mm}^2 = \text{VFEBTURE}(10) * (\text{pix. mm} ** 2) * \text{XY scale}$$

Switch-wi^{09-30R} Parametri

- VISION
- V. RECOGNITION [cam]
- V. BACKLIGHT [cam]
- V. BOUNDARY [cam]
- V. HOLES [cam]
- V. PERIMETER [cam] → fr. exterioră
- V. TOUCHING [cam]
- V. OVERLAPPING [cam]
- V. DISJOINT [cam]

-
- V. CENTROID [cam] - sist. calc. coord. centrului
de masă pt. ob. care apar
în scenă
 - V. 2ND. MOMENTS [cam] - calc. orient. obiectelor
înghiul între A și M₁ și M₂

09/3CR
PARAM. camm virtuelle



Ex: PARAMETER $\text{name-param} = \text{val}$

V. MIN. AREA [cam]

V. MAX. AREA [cam]

V. MIN. HOLE. AREA [cam]

} filtre

Ex:

V. GAIN [cam] $\rightarrow$ amplif. signal analogic

V. OFFSET [cam] $\rightarrow$ luminos. pixel arand mil de grau 1.

V. THRESHOLD [cam]

V. 2ND. THRESHOLD [cam]

C9/SC8

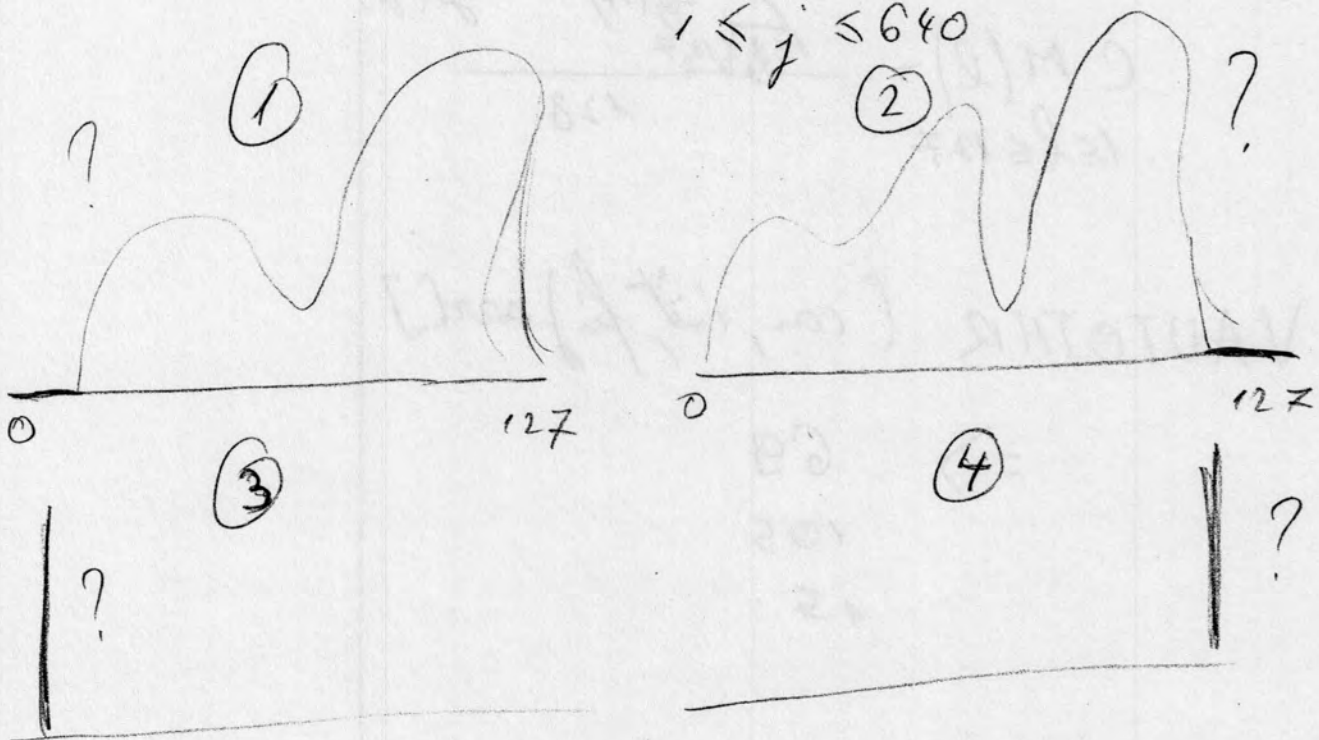
→ histograma imag. cu nivel de gri

$$hg: \{0, 1, \dots, 127\} \rightarrow \{0, 1, \dots, 480 \times 640\}$$

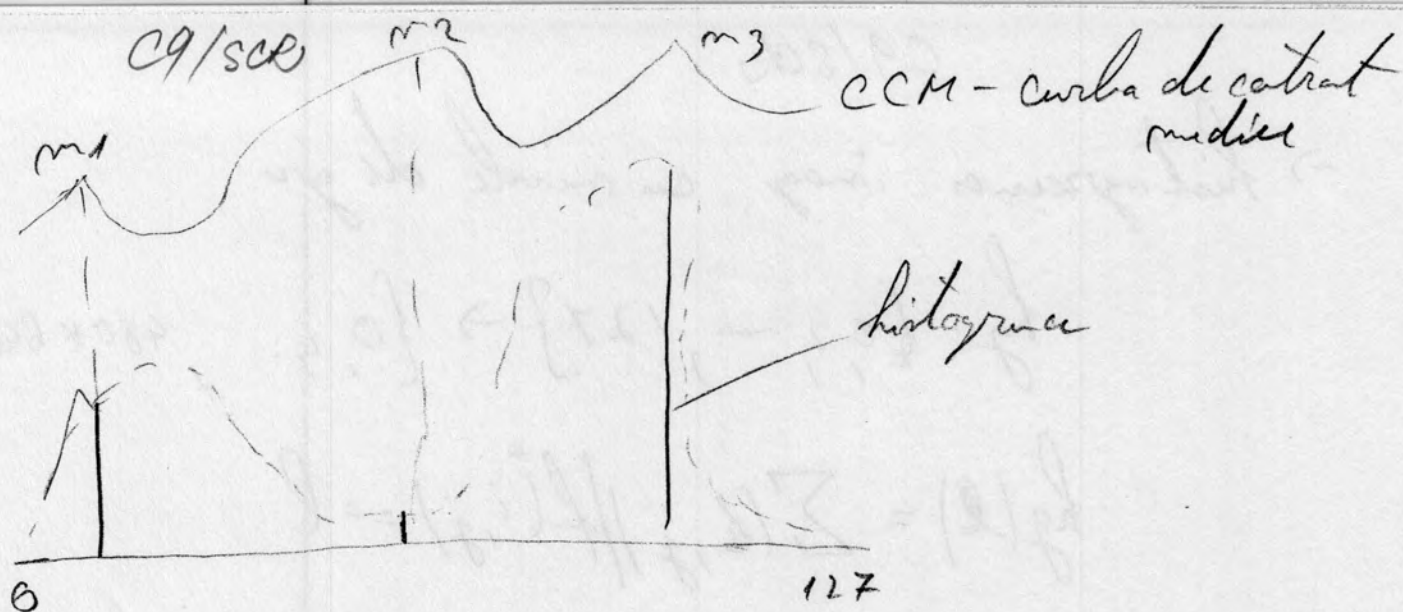
$$hg(l) = \sum_i (a_{i,j} / f^*(i,j)) = l$$

$$1 \leq i \leq 480$$

$$1 \leq j \leq 640$$



- (1) → n reach VOFFSET
- (2) → n crest VOFFSET
- (3) → n crest VGAIN
- (4) → n reach VGAIN



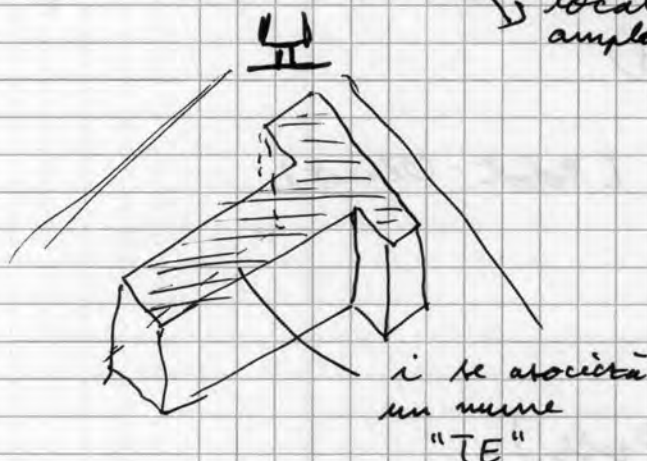
$$CM(l) = \frac{\sum_{1 \leq l \leq 127} h_g(l) - h_g(j)}{128}$$

VAUTOTHR (can, init, fin) arr[]

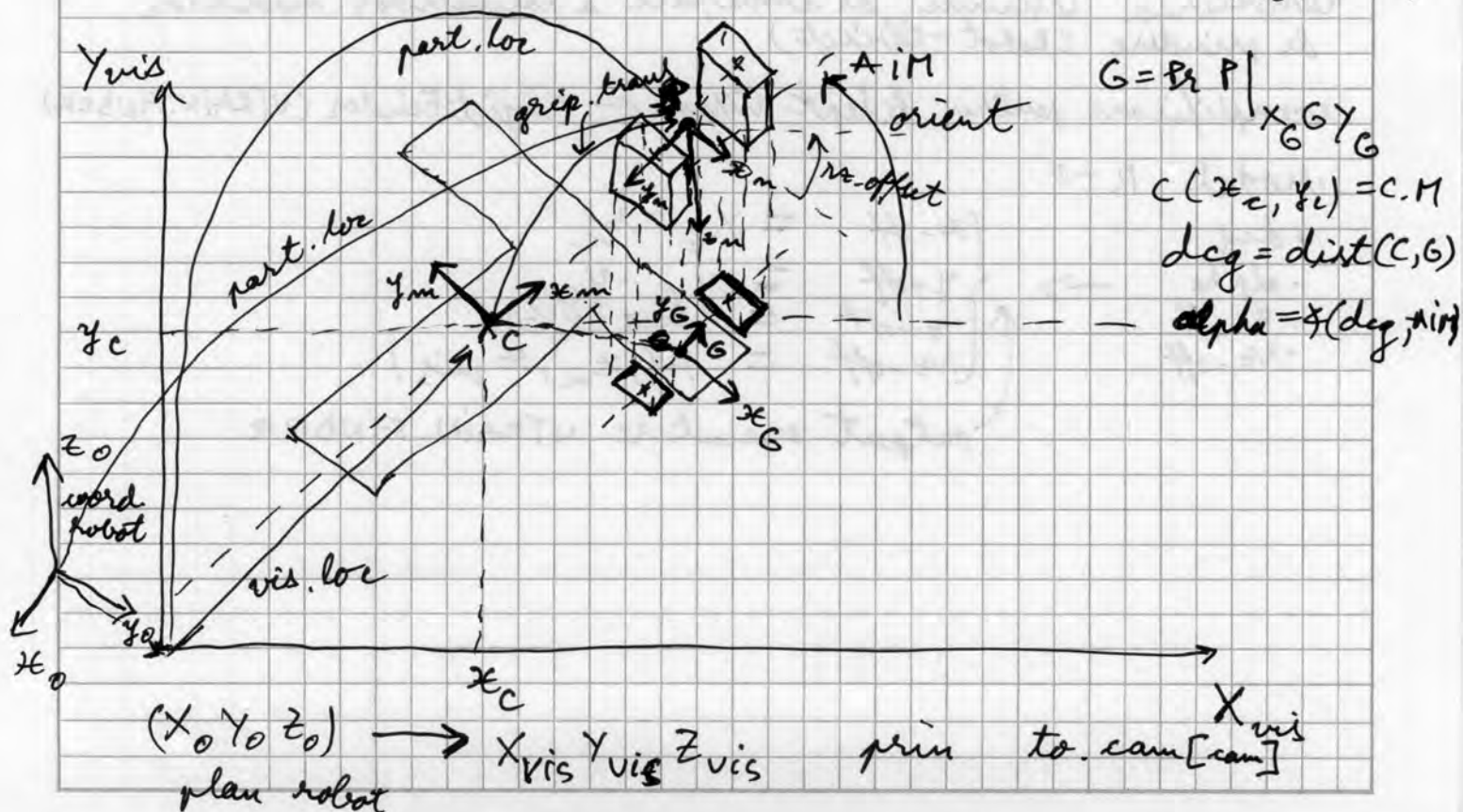
=> 69
105
15

Localizarea obiectelor

- are 2 componente

localizare obiect în plan imagine
(abstracție) $\Rightarrow$ vis. loclocalizare gripei relativ la proiecția
amplasamentului de prindere $\Rightarrow$ grip. transi se asociază
un nume
"TE"blob (în terminologie
"kdept, Vt, etc")
pata, imagine 2D, obiect
momentan, regiune x_m, y_m - sistem coord obiect $O_m x_m$ - se aliniază cu AIMgrip. trans $\Rightarrow$ model de
prindere
(model robot - obiect)C - centru de masă, (x_c, y_c)

AIM - axa de inerție minimă

orient = $\angle(Ox_{vis}, AIM)$  (x_0, y_0, z_0)
plan robot x_c $x_{vis} y_{vis} z_{vis}$

prin to. cam [cam]

SET part.loc = to, cam[cam] : vis.loc : grip.trans
 part.loc = amplasament robot ~~to~~ relativ la SR robot
 (x_0, y_0, z_0)
 part.vis = amplasament robot relativ la SR imagine
 ($x_{vis}, y_{vis}, z_{vis}$)

Parametrii modelului de prindere (Robot-Obiect)

→ deg

→ alpha

→ z-offset = (dist(P, G))

→ rz-offset = $\neq(x_m, x_{vis})$

deg, alpha, z-offset, rz-offset = $\neq(C, AIM, \text{"stil prindere"})$

VTRAIN.FINDER → învățare modelul de recunoaștere
 ↓
 mod = 14 sau 5? → inversează stilul de prindere

Descrierea schemei de învățare a parametrilor modelelor de prindere (Robot-Obiect)

exemplificare pentru Adept Vision ← ObjectFinder (VTRAIN.FINDER)

Model R-0

- deg
- alpha
- z-off
- rz-off

→ $\begin{cases} x\text{-off} &= x_G - x_C \\ y\text{-off} &= y_G - y_C \\ z\text{-off} &= \text{dist}(P, G) \\ rz\text{-off} &= \neq(x_m, x_{vis}) \end{cases}$
 output executare VTRAIN.FINDER

1. MCP $\rightarrow$ plasare manuală a robotului în amplasamentul dorit de prindere

$$\text{cam} = 1$$

2. HERE pt. role $\in$ imaginary self robot

3. DO OPENI

DO DEPARTS 100
DO MOVE outside

4. DO SET. to. cam[1]: pt. vis = pt. role ← se afla pt. vis - cu comp (x, y, z, Rx, Ry, Rz)

5. DO DECOMPOSE arr. pt[1] = pt. vis ← introduce cele 6 comp ale pt. vis in arr

6. $VPICTURE(\underline{1}, -1) - 1, \underline{1} \leftarrow$ how many
DO $VLOCATE(1, 2) "?", vis. loc [x_c, y_c, 0, \dots, 0] \leftarrow$ se returneara
in vis. loc
amplasamentul obiectului

7. DO DECOMPOSE arr.c[] = vis.loc

8. DO ~~X~~-off = arr.pt[0] - arr.c[0]

DO Y-off = arr.pt[1] - arr.c[1]

Z-off = arr.pt[2]

9. DO SET grip.par = TRANS (~~X~~-off, Y-off, Z-off, 0, 0, 0)

DO SET to.cam[1]: vis.loc: grip.par: rotz = pt.rot

DO DECOMPOSE arr.roll[] = rotz

DO RZ-off = arr.roll[5]

Mode R-O $\left. \begin{array}{l} - X\text{-off} \\ - Y\text{-off} \\ - Z\text{-off} \\ - RZ\text{-off} \end{array} \right\} \text{UTRAIN.FINDER}$

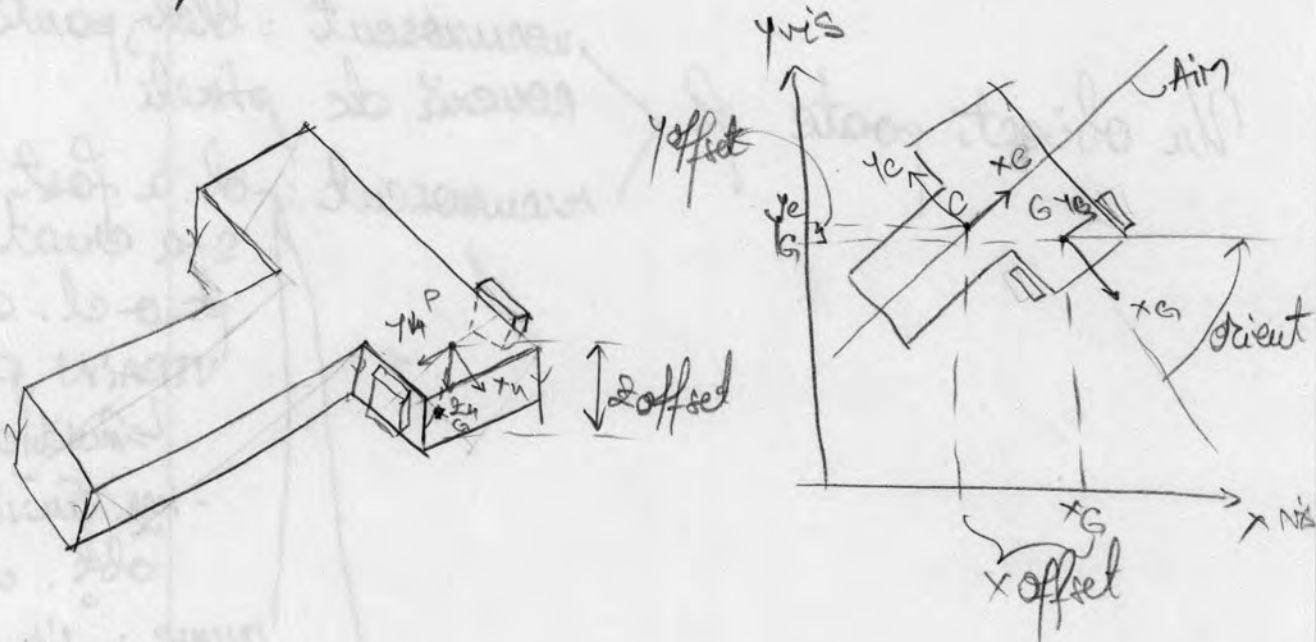
VLOCATE (cam, mode, order) \$name, vis.loc

verificată cu VFEATURE (1) - dacă VLOCATE s-a executat cu succes

Localizare de obiecte

VLOCATE (cam, mode, order) \$ name, vis. loc.

- fct. e doar instructiune de pg. ce ^{↑ x_o, y_o, orient}localizează piesele
- realizarea cu succes a ac. instr. permite localizarea
ob. în pl. imag. ⇒ obt. pozitie
de coord. ataseat robotului pe pl. imag. cf. unui stil
de prindere 'apriori' învârtite.



x_offset
 y_offset
 xz_offset
 z_offset

$\left. \begin{array}{l} x_offset \\ y_offset \\ xz_offset \\ z_offset \end{array} \right\} \text{VTRAIN.FINDER mode} = 5$

vis. loc. var. de tip transf. relativă din care 3 param. sunt 0.

$\begin{array}{c} \text{X Y Z R X R Y R Z} \\ \uparrow \quad \uparrow \quad \quad \uparrow \\ x_o \quad y_o \quad \quad \text{orient} \end{array}$

? = 0 pt. vedere pt. să vedem nu vede în gr.

- 2 se va utiliza din pick-place

a) Cam - id-ul cam. virtuală pe care noi o folosim
- p. o cam. fizică putem def. 2 cam. virtuale

Amoale = 3 biti

= B₂ B₁ B₀
MSB LSB

B₀ = $\begin{cases} 0, \text{wait} \\ 1, \text{no-wait} \end{cases}$

→ de. se impune ca aut progr.
să fie autorizat să fel
de. s-a terminat locali-
zarea sau nu.

B₁ = $\begin{cases} 0, \text{Final any} \\ 1, \text{Final particular} \end{cases}$

→ găsește (V) piesă sau
unde piesă particulară

Un obiect poate fi $\begin{cases} \text{recunoscut : glob, pantă, regiune} \\ \text{conexă de pixeli} \\ \text{recunoscut :} \end{cases}$

ob. a fost învățat
s-a creat un model
pt. o cl. de ob.

VTRAIN, FINDER, modes
↳ învățare offline
- reg. învățării se
ob. cu VFINDER

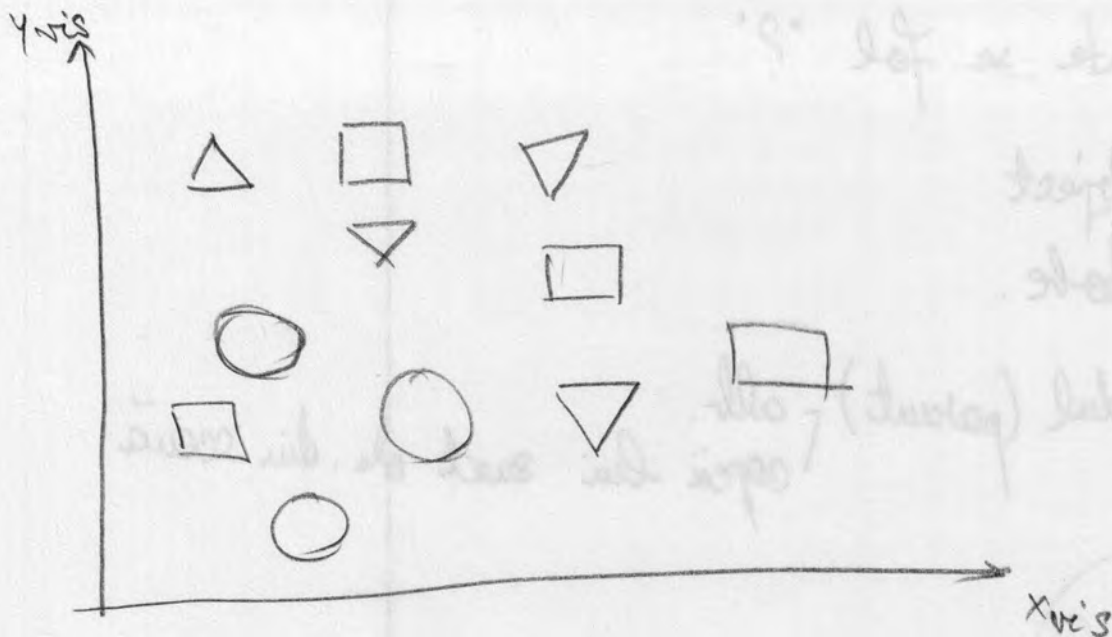
nume : - nr. de char.
"FLANSA 4"

a fost recunoscută o instanță a ob. (un model)

Că alternativă la învățarea ob. modelului este învățarea de caracteristici și discriminarea între ob.

▽ - triunghi
□ - pătrat
○ - disc

ob. din p. vederii → se dorește împărțirea pe
conținuturi de piese de ac. tip.



Find any - mod de localizare a (V) ob. recunoscut sau
reg. recunoscute care îndeplinesc cond. de
moment ORDER

Find particular - se localizează un ob. dintre-o cl, respectând
criteriile

→ implică \$name < any : (1) \$name - et. ce var. în
care sis. va returna o inf. despre ob. loc.
particular : "name" - et. ce reprez.
numele cl.

Find any → găsiți scene

```

VLOCATE ( ... ) $name,
IF VFEATURE(1) THEN
  IF $name == "PĂTRAT" THEN
    ≡ instr. ce tratează pătratul
  ELSE
    IF $name == "TRIUNGHI" THEN
      ≡ instr. ce tratează triunghiul
    ELSE
      IF $name == "DISC" THEN
        ≡ instr. ce tratează discul
      END
    END
  END

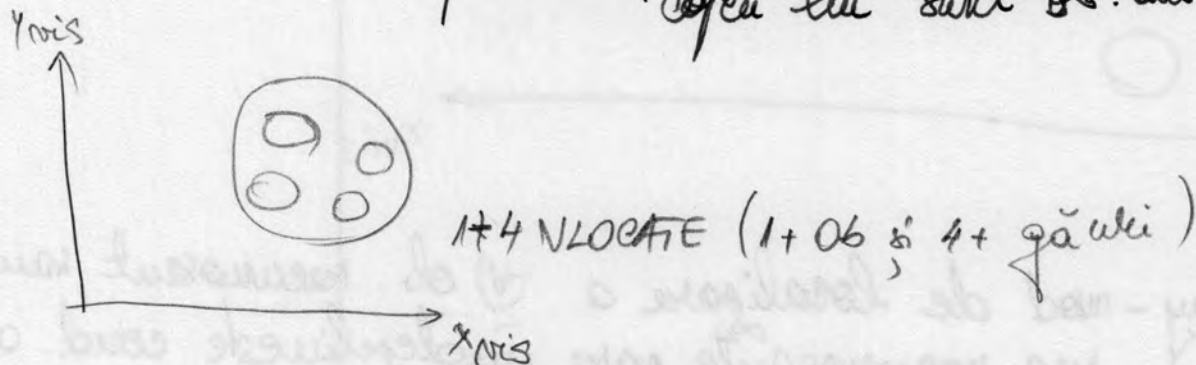
```

END END

- pt. ob. reînvițate se fol. "?"

-> B₂ { 0, obiect
1, hole.

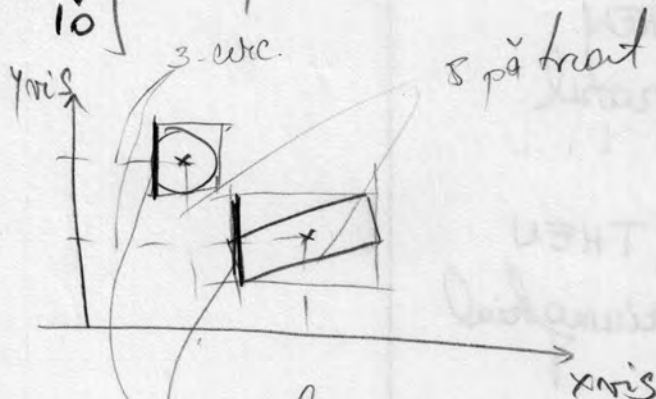
! imp. este fondul (parent) alb.
copii lui sunt ob. din scena



order - st. o ordine preferențială de localizare a ob.

- 0 -> aici o ordine și putem să ne d. seriem
 - 1 -> always the biggest
 - 2 -> always the smallest
 - 3 -> always the leftmost
 - 4 -> ——— rightmost
 - 5 -> ——— downmost
 - 6 -> ——— upmost
- } d.p. d.v. al C(x_c, y_c)

7
8
9
10 } la fel ca 3 ÷ 6 d.p. d.v. al dim
↳ de în considerare
v.ale cărui axe sunt
|| cu axele x_{vis}, y_{vis}
de arie min.



la 7-10 se măs. muchia îngroșată

Beau)

După calc. off-line al param. modelului de simulare
(x-offset, y-offset, rz-offset, z-offset), primii 3 treb. intro-
dus în sis. at. când reînnoașterea se face pe baza de mo-
dele învătate

VTRAIN.FINDER(cam, mode, dmode, arg, arg2, arg3)
\$model-name, ibt = val, val2, val3

- cam - nr. cam. virtuală
- mode - 2 moduri - 5 pt. introducerea celor 3 param.
= 14

- dmode - ignorat

- arg
arg2
arg3 } depind de mode

- \$model-name

- ibt = image buffer → ~~ex~~ subsp. a p. imag

- val ← x-offset
val2 ← y-offset
val3 ← rz-offset

- pt. mode = 14 → val / -5, 5 → derivatia AiM
val2

Recunoaşterea obiectelor

- 2 moduri după "trăsături": iniţiere a unei sesiuni de învăţare a trăsăturii pieselor.
după model → învăţare a modelului
→ VTRAIN.FINDER mode=3

1. Selecţie trăsăturilor care identifică cel mai bn. un ob.
(care) discriminează cel mai bn. o cl. de ob. de celelalte

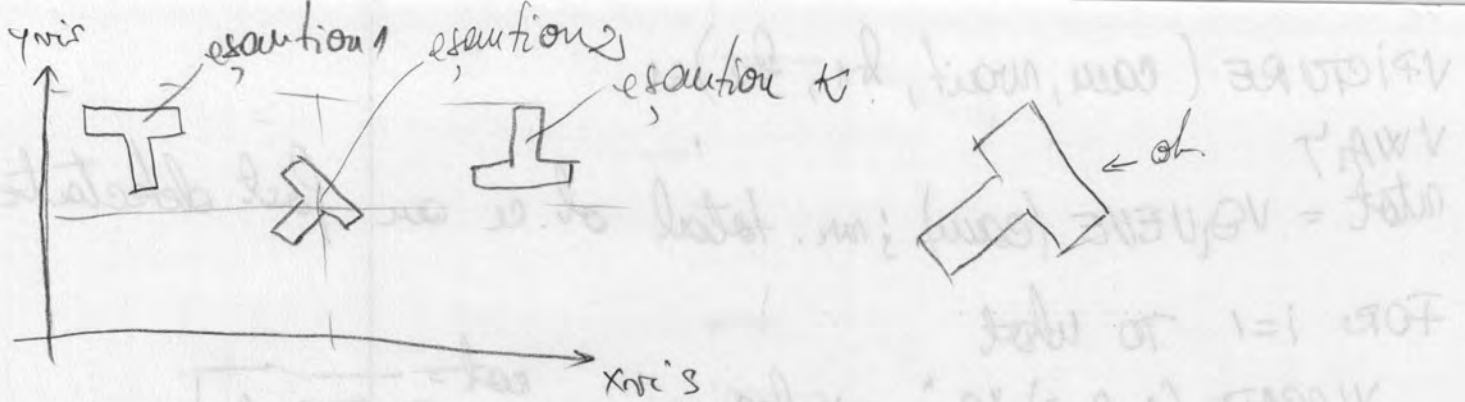
cl) $\left\{ \begin{array}{l} \text{aria} = S \\ \text{perimetru} = P \\ \text{nr. găuri} \\ \text{fact. de rotunjime} = r \end{array} \right.$

→ pt. un cerc: $\left\{ \begin{array}{l} S = \pi R^2 \\ P = 2\pi R \\ r = \frac{P^2}{4\pi S} \end{array} \right.$

Pt. fiecare reprezentant (cl)

exemplu măsuratură	1	2	3	n
arie	404	390	410	408
perimetru	184	186	179	182
nr. găuri	4	4	4	4
factor de rotunjime	0,96	0,95	0,96	0,98

→ cel puțin 6 ori



VPICTURE (cam, ^{wait}buf1, buf2) mode, 1

VF(10) - ar. - în pixeli
VF(41) - perim
VF(17) - nr. găuri

DO VLOCATE (1,2,3) "0", nr. loc.

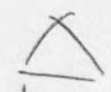
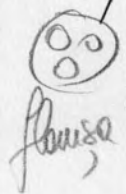
DO TYPE VFEATURE(1)

DO TYPE VFEATURE(10)

DO ar. mm = VFEATURE(10) * Yscale ** 2 * XY ratio

DO f. round = (VFEATURE(41) ** 2) / (4 * pi * ar. mm)

↑ - se returne cea mai mare val. (dintr-o de la fiecare înălțime) precum \$ min



un an model

ar. min. f = 390

ar. max. f = 410

per. min. f = 17.9

per. max. f = 18.4

nr. f = 4

frot min. f = 0.95

frot max. = 0.98

VPICTURE (cam, wait, b1, b2) - 1

VWAIT

ntot = VQUEUE (cam); nr. total ob. ce au fost detectate

FOR i=1 TO ntot

VLOCATE (1, 2, 3) "?", vis. loc.

IF VFEATURE (1) THEN

OK.f = ((ar >= ar.min) AND (ar <= ar.max)) AND

((per >= per.min) AND (per <= per.max)) AND

holes == 4 AND ((rot >= rot.min) AND (rot <= rot.max))

IF OK.f THEN

; strategia ob. de tip flausă

SET place

CALL pick-place()

nr.f = nr.f + 1

ELSE

OK.t = - - -

END

END

END

- tehnica se utilizează pt. ob. spl. ce se recunosc ușor

Turnăfarea de modelului

Recunoaștere după „model”

→ Object Finder (V+)

Etapă de parcurs

1. învățare model (creare/generare model) - offline
2. Planificare a modului în care se va efectua căutarea - offline
(identificarea)
3. identificare propriu-zisă obiecte (Search) - online

① Generare model

- constă în generarea unui model de recunoaștere pt fiecare clasă de obiecte
- model din imaginea cu niveluri de gri
- parametri → ai modelului
→ de căutare
- se face pe un singur cadru/sample
- 2 etape → 1: Detectare contururi
→ 2: Generare și reținere set de trăsături (se bazează pe contururi) → Edge features

Cerințe impuse:

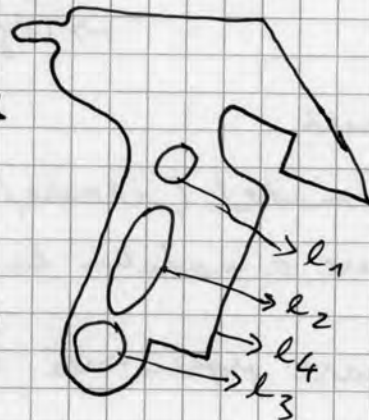
1. obiectul să fie reprezentativ pt clasa din care face parte (nu defecte, etc.)
→ în zona valorilor medii
2. imaginea să fie lipsită de zgomot
 - fond/fundal
 - obiecte
3. Fond uniform

Operații

1. Definire AOI → VDEF. AOI
area of interest
↓
ibr image buffer region
2. Multi-instrucțiunea VTRAIN.FINDER



Edge features
 → sistemul elimină
 contururi a false



se rețin → contururile închise
 → care au o lungime minimă impusă
 → acele contururi care au contrast bun
 între fore- și background

contururi atomice → linii drepte
 → arce de cerc

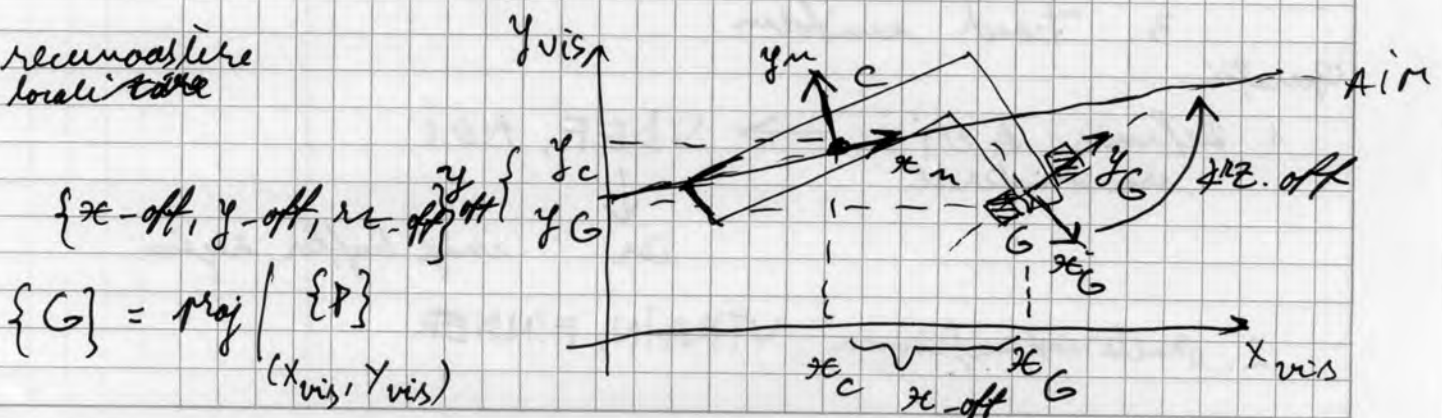
sistemul generează și fiecare contur o lungime
 ponderată: $L_{p_i} = \sum_{i=1}^n l_i \cdot w_i$

$|L_c - L_{p_i}| \leq E \cdot L_p$ $E = 1 - p$; $p = \text{verify percentage}$
 → lungime contur candidat

Etape pt verificarea concordanței

- Outline → reține un nr de contururi candidat
 → grobieră
- Detail → de detaliu

- recunoaștere
 - localizare



VTRAIN.FINDER (cam, mode, dmode, arg, arg2, arg3) \$ name-model, ibr=val, val2, val3

pt recunoastere : mode = 3 → def param de recunoastere asociati

↳ arg=3: defineste "verify percentage"

arg=10: defineste "recognition level"

arg, arg2, arg3 → valori > 0 ; def. param de search

→ valori < 0 ; def. param de model

ex : VTRAIN.FINDER(1, 3, ..., 3) ^{"FLANSA", etc} = 95 (0 - 100%)

VTRAIN.FINDER(1, 3, ..., 10) = 8 (1 - 10)

VTRAIN.FINDER(1, 3, ..., -15) = 0 {0, 1}

0 → lasa sistemul sa decida criteriile
pt filtrare / nivelurile de filtrare
la OUTLINE si detail

mode = 5 → pentru localizare

VTRAIN.FINDER(1, 5, ^{arg, arg2,} arg3) "L", ibr = X-off,
Y-off, rs-off

mode = 14 → pentru restrictii de
cantare

VTRAIN.FINDER(1, 14, ...) nume = alfamin, alfamax

alfamin, alfamax = # min/max dintre
A10 si X

→ restrictie de orientare

2. Planificare

VPLAN.FINDER (cam, mode, dmode, ambiguity)

\$ fmods[], \$ bmods[]

ambiguity → deține între corpuri cu detalii ce pot conduce la ambiguități

fmods → foreground models

→ tablou/listă de modele de obiecte învățate

bmods → background models

presupunem "FLANSA", "LIG", "TT" învățate

... \$ fmods[0] = "FLANSA"

\$ fmods[1] = "LIG"

\$ fmods[2] = "TT"

\$ fmods[3] = "" → capătul listei

→ când prog este rulat, 4 obiect va fi comparat și match-uit cu modele specificate

3. Recunoaștere (Search) → online

VFINDER (cam, mode, dmode, how-many) ibz

how-many → ~~număr~~ câte obiecte să recunoască

→ "-1" → oricât de multe

recunoaște este ~~VFINDER (cam, mode, b1, b2)~~

VPICTURE (cam, wait, b1, b2) mode

VFINDER (cam, mode, dmode, how-many) ibz

VWAIT

mode
-1, 0, 1, 2
→
2 = achiziție only

nt obiecte fără model de recunoaștere, sistemul le da un nume default: "?"

Parametrii pt Search

1. Contour detection parameters

OUTLINE (1-10)
DETAIL (1-10)

UTRAIN.FINDER (-, 3, --, -15) = 0 → automat

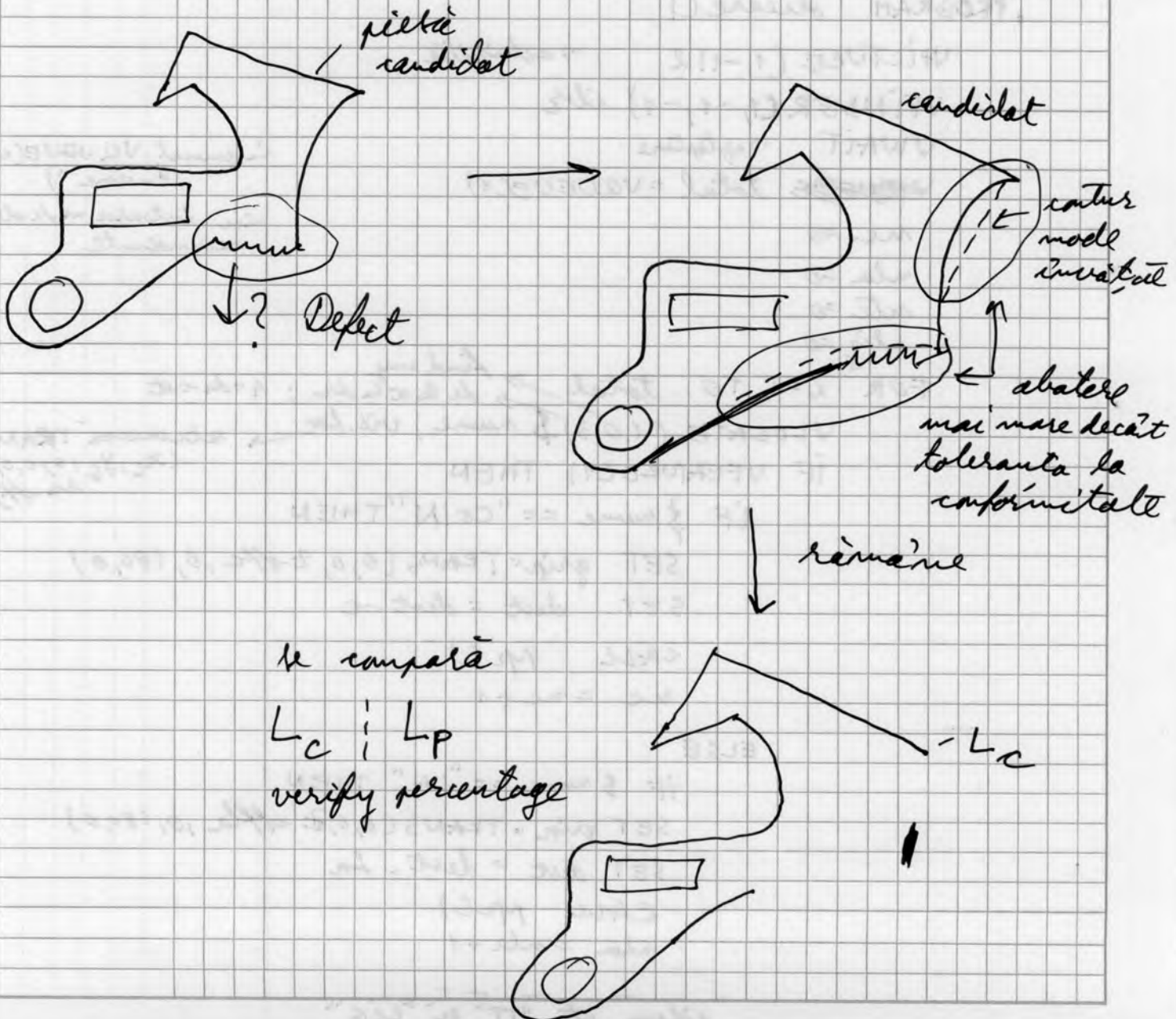
2. Contrast parameters

3. Search constraints (orientarea AIM) ↗ alfa min ↘ alfa max

4. Recognition level (1-10)

5. Conformity tolerance

6. Verify percentage



Program

- Am învățat un nr de 4 modele: "COIN", "LA", "TE", "LIG"
 se dorește mutarea lor la: dest-c dest-la dest-te dest-lig
 pt fiecare destinație avem x-off, y-off, z-off, rc-off
 distanța de pick and place: depi, depl
 așteptare de la comandă deschidere gripper până la
 urm. comandă: 0,5 sec
 viteză maximă 100% cu excepția pick, place → 25%
 punct lape învățat
 se lucrează cu numere virtuale

PROGRAM pricare()

VPICTURE (1,-1)2 → achiziție

VFINDER (1,-1,-1) ilr

VWAIT → așteptare

~~VQUEUE~~ total = VQUEUE(1)

mc = 0

mca = 0

mte = 0

mlig = 0

FOR i=1 TO total → find any de la st la dr; 4=dr→st

VLOCATE (1,0,3) \$nume, vis.loc → returnarea TRANS (x₆, y₆, 10) 0,0,0,0,0,0
 IF VFEATURE(1) THEN

IF \$nume == "COIN" THEN

SET grip = TRANS(0,0,z-offc,0,180,0)

SET dest = dest-c

CALL prc()

mc = mc + 1

ELSE

IF \$nume == "LA" THEN

SET grip = TRANS(0,0,z-offla,0,180,0)

SET dest = dest-la

CALL prc()

mca = mca + 1

idem pt "TE" și "LIG"

```

END      ← pt if VFEATURE
END ← pt for
MOVE rate
END

```

```

PROGRAM MR()
PARAMETER HAND.TIME = 0.5
SPEED 100 ALWAYS
SET pick = to.cam[1]:vis.loc: grip
APPRO pick, dspr
BREAK
SPEED 25
MOVES pick
CLOSE i
DEPART dspr
APPRO dest, dspr
BREAK
SPEED 25
MOVES dest
OPEN i
DEPARTS dspr
RETURN
END

```

2. Acelari lucru, intra grupat pe tipurile pe picte

```

PROGRAM ordline()
VPICTURE (1,-1) 2
VFINDER (1,-1,-1) itur
VWAIT
totc = VQUEUE(1, "coin")
totla = VQUEUE(1, "LA")
totte = VQUEUE(1, "TE")

```

totlig = NQUEUE(1, "LIG")

mc = 0

nlig = 0

FOR i=0 TO totc *→ "lind-particular"*

VLOCATE(1, 2, 3) "COIN", ves.loc

IF VFEATURE(1) THEN

SET dest = dest.c

SET grip = TRANS(0, 0, 2, 0, 180, 0)

CALL nnc()

mc = mc + 1

END

END

--- for script relative figure de pick

MOVE safe

• END