Prepoznavanje kružnica i elipsi

Nikić, Patrick

Sveučilište J.J.Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike

Patrick Nikić

Prepoznavanje kružnica i elipsi

Završni rad

Osijek, 2017.

Sveučilište J.J.Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike

Patrick Nikić

Prepoznavanje kružnica i elipsi

Završni rad

Voditelj: prof.dr.sc. Rudolf Scitovski

Osijek, 2017.

Sažetak. Cilj ovoga završnoga rada je dati pregled osnovnih obilježja elipse i opisati odabrane algoritme za prepoznavanje kružnica i elipsi u podacima i na slikama. Opisat ćemo inačicu k-means algoritma za prepoznavanje kružnica i protumačiti kako se elipsa može interpretirati kao Mahalanobis kružnica sa nekom pozitivno definitnom matricom. Baviti ćemo se problemom prepoznavanja više elipsi u ravnini na temelju podataka koji dolaze iz skupa elipsi čiji broj nije unaprijed poznat. Biti će opisan algoritam za prepoznavanje elipsi koji se temelji na statistčkom modelu elipse. Nadalje, u svrhu efikasne implementacije prethodnog algoritma, predstavit ćemo konstrukciju C++ programa istoga. Konačno, testirati ćemo program na različitim generiranim primjerima i primjerima iz svakodnevnog života.
Ključne riječi: prepoznavanje kružnica, k-means, prepoznavanje elipse, RANSAC

Abstract. The aim of this final thesis is to provide an overview of the basic characteristics of the ellipse and to describe selected algorithms for circle and ellipse detection from data and images. We will describe a version of the k-means algorithm for circle detection and explain how the ellipse can be considered as a Mahalanobis circle with some positive definite matrix. We will address the problem of detecting multiple ellipses in the plane based on the data coming from a set of ellipses whose number is not known in advance. An algorithm for ellipse detection, which relies on a statistical model of the ellipse, will be described. Furthermore, in order to efficiently implement the previous algorithm, we will present a C++ implementation of it. Finally, we will test the program on different generated examples and examples in everyday life.
Keywords: circle detection, k-means, ellipse detection, RANSAC

Sadržaj

1 Uvod

Prepoznavanje kružnih objekata i elipsi u digitalnim slikama učestali je problem u obrađivanju slika. Problem prepoznavanja elipsi javlja se u različitim područijima znanosti kao što su računalni vid i prepoznavanje uzoraka [5], medicina, robotika i drugim primjenama.

U ovom radu bavit ćemo se sljedećim problemom:
Neka su (a_1, a_2) i $(b_1, b_2) \in \mathbb{R}^2$ dvije točke u ravnini koje definiraju pravokutnik $\mathcal{P} = \{(x, y) \in \mathbb{R}^2 : a_1 \leq x \leq b_1, a_2 \leq y \leq b_2\} \subset \mathbb{R}^2$ i $\mathcal{A} = \{a^i = (x_i, y_i) : i = 1, \ldots, m\} \subset \mathcal{P}$ skup podataka koji dolazi iz skupa elipsi u ravnini koje nisu unaprijed poznate. Za dani skup $\mathcal{A}$, potrebno je odrediti te elipse. Postoje različiti pristupi ovome problemu. Može se formulirati optimizacijski problem, pri čemu primjerice minimiziramo udaljenost skupa točaka do skupa elipsi ili prepoznavati elipse Houghovom transformacijom. Navedeni pristupi su, u pravilu, teško primjenjivi u realnom vremenu. Akinlar i Topal [1] su predložili algoritam za prepoznavanje kružnica i elipsi bliskih kružnicama koji radi u realnom vremenu. Prasad, Leung i Quek [6] predlažu neiterativnu metodu kojoj je cilj prepoznavanje elipsi samo kada je vrlo vjerojatno da podaci dolaze sa elipse.

Cilj je ovog rada prezentirati algoritam za prepoznavanje kružnica i elipsi u ravnini na osnovi podataka mjerenja, pri čemu rubovi elipse ne moraju biti precizni ili jasni. Broj elipsi nije unaprijed zadan i one se mogu nalaziti u međusobno različitim položajima tj. biti razdvojene ili se presijecati. Riječ je o algoritmu koji su predložili Grbić, Grahovac i Scitovski [4].

Rad je podijeljen na nekoliko poglavlja. U drugom poglavlju ćemo dati osnovne oblike jednadžbe elipse i pokazati kako iz njih isčitati obilježja elipse. U nastavku ćemo spomenuti poseban slučaj problema prepoznavanja elipsi tj. problem prepoznavanja kružnica i kako elipsu možemo shvatiti kao Mahalanobis kružnicu. U petom poglavlju ćemo opisati rad spomenutog algoritma za prepoznavanje kružnica i elipsi. Nakon toga ćemo predstaviti konstrukciju programa na bazi prethodnog algoritma i izložiti rezultate testiranja istog.

2 Obilježja elipse

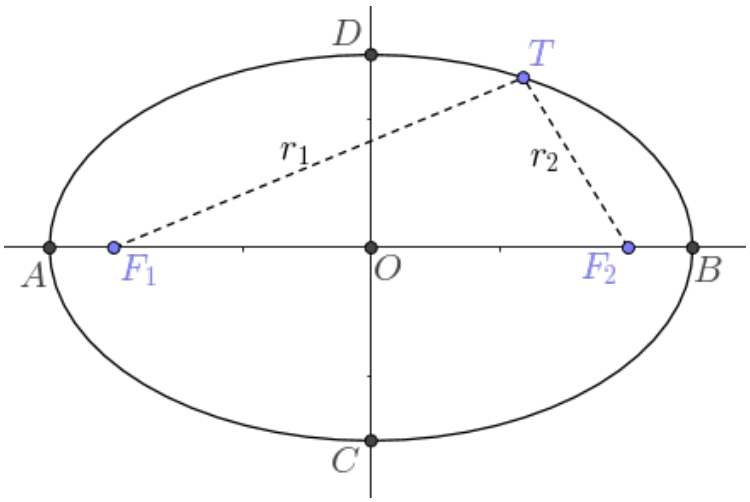
Uvodeći konike u nastavi matematike, često se navode primjeri iz stvarnog života. Neki od njih su: odsjaj svjetiljke na zidu za hiperbolu, putanja gibanja baseball loptice za parabolu, Trg svetog Petra i Kolosej u Rimu za elipse. Pošto je riječ o objektima koji su čovjeku poznati od davnina, rabe se različiti nazivi za konike. Stari Grci su ih definirali kao presjeke stošca ravninom, pa se u hrvatskom jeziku nazivaju i čunjosječnice.

Definicija 2.1. ***Konika*** *ili* ***krivulja drugog reda*** *je skup svih točaka ravnine koje zadovoljavaju sljedeću algebarsku jednadžbu:*

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0.$$

Koeficijenti A, B, C, D, E i F su proizvoljni realni brojevi takvi da su barem jedan od A, B ili C te barem jedan od D, E ili F različiti od nule. Uočimo da se sve konike mogu prikazati gornjom jednadžbom, ali neki izbori koeficijenata mogu dati tzv. degenerirane konike (točka, pravac, ukriženi pravci), paralelne pravce ili čak prazan skup. Najčešće će gornja jednadžba dati nedegeneriranu koniku: elipsu, hiperbolu ili parabolu. U ovom radu će se razmatrati elipse. Navodimo najprije geometrijsku definiciju elipse.

Definicija 2.2. *Neka su F_1 i F_2 dvije zadane točke ravnine i neka je a pozitivan realan broj, $a > \frac{1}{2}|F_1F_2|$. Skup svih točaka ravnine za koje je zbroj udaljenosti do točaka F_1 i F_2 jednak $2a$ nazivamo* ***elipsa****.*



Slika 1: Elipsa

Točke F_1 i F_2 nazivaju se žarišta elipse. Polovište O dužine F_1F_2 naziva se središte elipse. Polovica udaljenosti između žarišta, $e = \frac{1}{2}|F_1F_2|$ naziva se linearni ekscentritet elipse.
Dužina koja spaja bilo koju točku T elipse s jednim od žarišta naziva se radijvektor točke T.
Dužina AB naziva se velika os elipse, pa su dužine OA i OB velike poluosi elipse. Duljina velike osi jednaka je $2a$ pa je duljina velike poluosi jednaka a. Pravac koji prolazi središtem okomito na veliku os siječe elipsu u točkama C i D. Dužina CD naziva se mala os elipse, pa su dužine OC i OD male poluosi elipse. Duljina male poluosi označava se s b.

Smjestimo li elipsu u pravokutni koordinatni sustav u ravnini tako da se smjerovi x i y osi podudaraju sa smjerovima velike i male osi elipse, za točku $T = (x, y)$ na elipsi, prema definiciji elipse, mora vrijediti

$$\sqrt{(x+e)^2 + y^2} + \sqrt{(x-e)^2 + y^2} = 2a.$$

Sređivanje gornjeg izraza daje standardni oblik jednadže elipse

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1.$$

Standardni oblik jednadžbe elipse u parametriziranom obliku je:

$$x = a\cos(t), \quad y = b\sin(t), \quad t \in [0,\, 2\pi). \tag{1}$$

Površina elipse u standardnom obliku dana je s:

$$P = \pi ab.$$

Za opseg elipse ne postoji izvedeni algebarski izraz. Potrebno je riješiti eliptički integral druge vrste, koji se u pravilu rješavaju numeričkom integracijom.
U nastavku navodimo definiciju elipse u općem obliku.

Definicija 2.3. *Elipsa u ravnini se definira kao skup* $E = \{(x,\, y) \in \mathbb{R}^2 : P(x,\, y) = 0\}$, *gdje je*

$$P(x,\, y) = Ax^2 + Bxy + Cy^2 + Dx + Ey + F, \quad 4AC - B^2 > 0.$$

Iz općeg oblika jednadžbe elipse ne mogu se izravno vidjeti njezina obilježja. Središte $(p,\, q)$ elipse dobivamo rješavanjem sustava jednadžbi

$$\frac{\partial P}{\partial x} = 0 \qquad\qquad \frac{\partial P}{\partial y} = 0.$$

Nakon jednostavnog računa dobivamo

$$p = \frac{BE - 2CD}{4AC - B^2} \qquad\qquad q = \frac{BD - 2AE}{4AC - B^2}.$$

Općenito, ne poznajemo orijentaciju elipse. Imajući na umu da je lakše računati s elipsom čije se osi podudaraju sa smjerovima osi koordinatnog sustava, tražimo kut θ za koji treba rotirati elipsu kako bi dobili standardnu orijentaciju. Tražimo rotaciju koja će nam to omogućiti. Primjenom sljedeće transformacije koordinata:

$$x = \widetilde{x}\cos\theta + \widetilde{y}\sin\theta$$
$$y = -\widetilde{x}\sin\theta + \widetilde{y}\cos\theta$$

dobivamo novu jednadžbu:

$$A_1\widetilde{x}^2 + B_1\widetilde{x}\widetilde{y} + C_1\widetilde{y}^2 + D_1\widetilde{x} + E_1\widetilde{y} + F_1 = 0.$$

Poštedimo li čitatelja tehničkih algebarskih operacija, koeficijenti su dani s:

$$A_1 = \frac{A+C}{2} + \frac{A-C}{2}\cos 2\theta - \frac{B}{2}\sin 2\theta \qquad D_1 = D\cos\theta - E\sin\theta$$
$$B_1 = \frac{A-C}{2}\sin 2\theta + B\cos 2\theta \qquad E_1 = D\sin\theta + E\cos\theta$$
$$C_1 = \frac{A+C}{2} - \frac{A-C}{2}\cos 2\theta + \frac{B}{2}\sin 2\theta \qquad F_1 = F.$$

Izjednačimo li B_1 s nulom dobivamo da je traženi kut θ rješenje jednadžbe

$$\tan 2\theta = \frac{B}{C-A}.$$

Za $A \neq C$, ova jednadžba ima četiri rješenja u $[0,\, 2\pi)$ koja se razlikuju za $\pi/4$:

$$\theta = \frac{1}{2} \arctan \frac{B}{C - A},$$

kojima odgovaraju četiri rotacije elipse do standardnog vodoravnog ili okomitog položaja. Za $A = C$, $\tan 2\theta$ nije definiran. U tom slučaju je $2\theta = \frac{\pi}{2}$ tj. $\theta = \frac{\pi}{4}$.

Nakon rotacije u standardnu orijentaciju, preimenujemo li koeficijente, jednadžba elipse glasi:

$$Ax^2 + Cy^2 + Dx + Ey + F = 0.$$

Translatiramo li središte elipse u ishodište koordinatnog sustava sljedećom transformacijom koordinata:

$$\widetilde{x} = x + \frac{D}{2A}$$
$$\widetilde{y} = y + \frac{E}{2C}$$

dobivamo

$$A\widetilde{x}^2 + C\widetilde{y}^2 - \frac{D^2}{4A} - \frac{E^2}{4C} + F = 0.$$

Došli smo do standardnog oblika jednadžbe elipse, iz kojeg možemo direktno vidjeti duljine velike i male poluosi:

$$a = \sqrt{\frac{\frac{D^2}{4A} + \frac{E^2}{4C} - F}{A}} \qquad\qquad b = \sqrt{\frac{\frac{D^2}{4A} + \frac{E^2}{4C} - F}{C}}.$$

3 Prepoznavanje kružnice

Poseban slučaj problema prepoznavanja elipsi je problem prepoznavanja kružnica. Pretpostavimo da skup podataka $\mathcal{A} = \{a^i = (x_i,\, y_i) : i = 1, \ldots, m\} \subset \mathbb{R}^2$ dolazi iz skupa kružnica u ravnini koje je potrebno rekonstruirati ili prepoznati.

Za rješavanje ovog problema Scitovski i Marošević [7] predlažu algoritam utemeljen na grupiranju podataka. Skup $\mathcal{A}$ treba podijeliti u k nepraznih, disjunktnih podskupova $\pi_1, \ldots, \pi_k$, $1 \leq k \leq m$, takvi da

$$\bigcup_{i=1}^{k} \pi_i = \mathcal{A}, \quad \pi_r \cap \pi_s = \emptyset,\, r \neq s, \quad |\pi_j| \geq 1,\, j = 1, \ldots, k.$$

Ovakvu particiju ćemo označiti s $\Pi(\mathcal{A}) = \{\pi_1, \ldots, \pi_k\}$, a njezine elemente $\pi_1, \ldots, \pi_k$ nazivamo clusterima. Svakom clusteru $\pi_j \in \Pi(\mathcal{A})$ odgovara kružnica $C_j(S_j,\, r_j)$ sa središtem $S_j = (p_j,\, q_j)$ i polumjerom r_j, koju povezujemo s sljedećim problemom globalne optimizacije

$$(p_j,\, q_j,\, r_j) = \operatorname*{argmin}_{p,q,r \in \mathbb{R}} \sum_{a^i \in \pi_j} D(C(p,\, q,\, r),\, a^i),$$

gdje $D(C(p,\, q,\, r),\, a^i)$ predstavlja udaljenost točke a^i do kružnice C (Vidi 3.2).

Ukoliko je funkcija cilja $\mathcal{F} : \mathcal{P}(\mathcal{A};\, m,\, k) \to \mathbb{R}_+$ definirana na skupu svih particija $\mathcal{P}(\mathcal{A};\, m,\, k)$ skupa $\mathcal{A}$ sa točno k clustera, možemo ocijenjivati particije i tražiti rješenje problema globalno optimalne k-particije rješavajući sljedeći problem globalne optimizacije

$$\operatorname*{argmin}_{\Pi \in \mathcal{P}(\mathcal{A};\, m,\, k)} \mathcal{F}(\Pi), \quad \mathcal{F}(\Pi) = \sum_{j=1}^{k} \sum_{a^i \in \pi_j} D(C_j(p_j,\, q_j,\, r_j),\, a^i).$$

Slično, za dani skup različitih kružnica $C_1, \ldots, C_k \subset \mathbb{R}^2$ može se definirati particija $\Pi = \{\pi_1, \ldots, \pi_k\}$ na skupu $\mathcal{A}$ principom minimalne udaljenosti na sljedeći način:

$$\pi_j = \{a \in \mathcal{A} : D(C_j,\, a) \leq D(C_s,\, a),\, \forall s = 1, \ldots, k,\, s \neq j\}, \quad j = 1, \ldots, k, \tag{2}$$

gdje treba obratiti pozornost na činjenicu da se svaki element skupa $\mathcal{A}$ nalazi u jednoj i samo jednoj particiji. Stoga se problem pronalaska optimalne particije skupa svodi na

$$\operatorname*{argmin}_{C_1, \ldots, C_k \subset \mathbb{R}^2} \mathcal{F}(C_1, \ldots, C_k), \quad \mathcal{F}(C_1, \ldots, C_k) = \sum_{i=1}^{m} \min_{j=1,\ldots,k} D(C_j,\, a^i).$$

3.1 Algoritam

Bit će korištena inačica k-means algoritma za traženje lokalno optimalne particije, pri čemu su kružnice središta clustera. Algoritam se može opisati u dva međusobno alternirajuća koraka:

A. Za dani skup međusobno različitih kružnica $C_1, \ldots, C_k$, skup $\mathcal{A}$ podijeliti u k disjunktnih, nepraznih clustera $\pi_1, \ldots, \pi_k$ koristeći princip minmalne udaljenosti (2).

B. Za danu k-članu particiju $\Pi = \{\pi_1, \ldots, \pi_k\}$ skupa $\mathcal{A}$ definirati pripadne kružnice središta clustera $C_1(p_1,\, q_1,\, r_1), \ldots, C_k(p_k,\, q_k,\, r_k)$ rješavajući sljedeće probleme globalne optimizacije

$$(p_j,\, q_j,\, r_j) = \operatorname*{argmin}_{p,q,r \in \mathbb{R}} \sum_{a^i \in \pi_j} D(C(p,\, q,\, r),\, a^i), \quad j = 1, \ldots, k. \tag{3}$$

Poznavajući dobre početne aproksimacije kružnica, ovaj algoritam daje prihvatljiva rješenja. U slučaju da početne aproksimacije nisu poznate, algoritam se može pokrenuti više puta sa različitim početnim aproksimacijama.

3.2 Udaljenost točke do kružnice

Za rješavanje problema prepoznavanja kružnice, vrlo je važno dobro definirati udaljenost točke do kružnice. Ta udaljenost koristiti će se pri primjeni principa minimalne udaljenosti (2) za pronalaženje kružnica središta clustera u (3). Neki od čestih izbora su:

$$D_1(C(S,\, r),\, a^i) = |\|S - a^i\| - r|,$$
$$D_2(C(S,\, r),\, a^i) = (\|S - a^i\| - r)^2,$$
$$D_3(C(S,\, r),\, a^i) = (\|S - a^i\|^2 - r^2)^2.$$

Posljednja udaljenost se naziva algebarska i često se koristi u primjenama. Algebarska udaljenost korisiti se i u [7] gdje možete pronaći više informacija o problemu prepoznavanja kružnica.

4 Elipsa kao Mahalanobis kruznica

Poznato je da je Euklidska udaljenost dviju točki $x, y \in \mathbb{R}^2$ dana s

$$\|x - y\| = \sqrt{(x-y)^T \mathbb{I}(x-y)},$$

gdje T označava operaciju transponiranja, a $\mathbb{I}$ pripadnu jediničnu matricu. Jasno slijedi da sve točke koje imaju jednaku udaljenost do ishodišta $\|x - 0\| = c$ zadovoljavaju $x_1^2 + x_2^2 = c^2$. Zaključujemo da obje komponente x_1 i x_2 jednako doprinose Euklidskoj udaljenosti točke x do ishodišta. U nekim slučajevima, poželjno je da komponente imaju različite težine. To možemo postići koristeći Mahalanobis udaljenost definiranu za pozitivno definitne matrice $\Sigma \in \mathbb{R}^{2\times 2}$ sa

$$\|x - y\|_\Sigma = \sqrt{(x-y)^T \Sigma^{-1}(x-y)}.$$

Elipsa sa središtem $c = (p, q)$, duljinama poluosi a i b, te kutom θ između poluosi a i pozitivnog dijela osi abscissa, se može interpretirati kao jedinična Mahalanobis kružnica (M-kružnica) sa središtem $c = (p, q)$

$$E(c, 1; S) = \{u \in \mathbb{R}^2 : d_M(u, c; S) = 1\},$$

gdje je $d_M : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}_+$ Mahalanobis udaljenost dana s

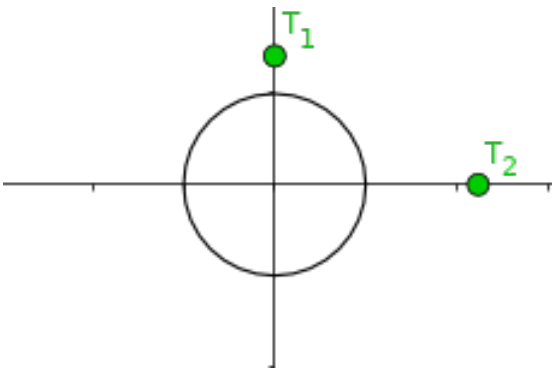
$$d_M(u, v; S) = \|u - v\|_S^2 = (u-v)^T S^{-1}(u-v),$$

a $S \in \mathbb{R}^{2\times 2}$ simetrična, pozitivno definitna matrica oblika

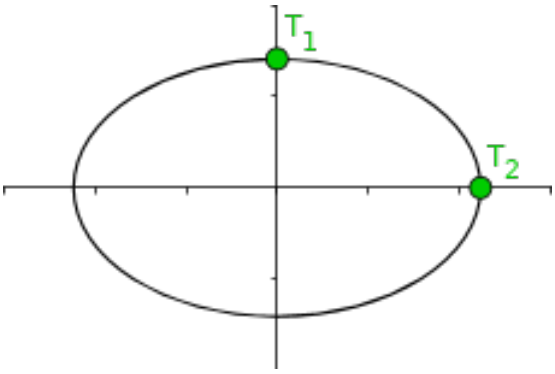
$$S = U \begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix} U^T, \quad \text{gdje je } U = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}. \tag{4}$$

Više o ovome može se pročitati u [4] i referencama unutar.

Primjer 4.1. *Slika 2 prikazuje jediničnu kružnicu* $\{u \in \mathbb{R}^2 : \|u\|_2^2 = 1\}$ *sa središtem u* $O = (0, 0)$*, dok slika 3 prikazuje jediničnu M-kružnicu sa središtem u* O *i pozitivno definitnom matricom* $S = \left[\begin{smallmatrix} 5 & 0 \\ 0 & 2 \end{smallmatrix}\right]$*. Neka su* $T_1 = (0, \sqrt{2})$ *i* $T_2 = (\sqrt{5}, 0)$ *točke u ravnini. Uočimo da su euklidske udaljenosti točaka* T_1 *i* T_2 *do ishodišta, redom,* $d(O, T_1) = 2$ *i* $d(0, T_2) = 5$*, ali obje točke leže na pripadnoj jediničnoj M-kružnici.*



Slika 2: Jedinična kružnica



Slika 3: Jedinična M-kružnica

5 Prepoznavanje elipse

Vratimo se problemu prepoznavanja elipse. Neka je $\mathcal{A} = \{a^i = (x_i, y_i) \in \mathbb{R}^2 : i = 1, \ldots, m\}$ skup točaka u ravnini koje dolaze iz skupa elipsi koje nisu unaprijed poznate. Po uzoru na poglavlje 3, poznavajući skup $\mathcal{A}$ i broj elipsi k, može se konstruirati inačica k-means algoritma za traženje lokalno optimalne particije $\Pi = \{\pi_1, \ldots, \pi_k\}$ skupa $\mathcal{A}$, pri čemu su središta clustera M-kružnice $E_1(c_1, r_1; S_1), \ldots, E_k(c_k, r_k; S_k)$. Za rad ovog algoritma je važno poznavati broj elipsi i dobre početne aproksimacije istih. Međutim, takav algoritam možemo primijenti na elipsama koje daje Algoritam 1 koji ćemo predstaviti u ovom poglavlju. Slijedi algoritam koji su predložili Grbić, Grahovac i Scitovski [4].

Nastavno na ideju RANSAC[1] algoritma, na svakoj iteraciji odabire se slučajna elipsa iz skupa slobodnih točaka. Za svaku takvu elipsu provjeramo dolazi li ona iz skupa elipsi koje su generirale skupa $\mathcal{A}$. Algoritam 1 obavlja N pokušaja. U svakoj iteraciji odabire slučajnih m_e točaka (x_j, y_j), $j = 1, \ldots, m_e$, i pronalazi odgovarajuću elipsu.

Ulaz: $\mathcal{S} \subseteq \mathcal{A}$; N; $m_e \geq 5$; p_{min};
Izlaz: $\pi^{(0)}$, $E^{(0)}$

$\pi^{(0)} \leftarrow \emptyset$
$E^{(0)} \leftarrow E(0, 0; I)$
za $s \leftarrow 1$ **do** N **čini**
  Slučajno odaberi m_e točaka $(x_i, y_i) \in \mathcal{S}$, $i = 1, \ldots, m_e$ i koristeći dekompoziciju matrice na singularne vrijednosti (SVD), u smislu najmanjih kvadrata odredi rješenje sustava $Ax_i^2 + Bx_iy_i + Cy_i^2 + Dx_i + Ey_i + F = 0, \quad i = 1, \ldots, m_e$
  ako $4AC - B^2 > 0$ **onda**
    Koristeći izraze izvedene u poglavlju 2 izračunaj parametre (p, q, θ, a, b) elipse $E^{(1)}$
    za $j \leftarrow 1$ **do** m **čini**
      $(u_j, v_j)^T \leftarrow U^T((x_j, y_j)^T - (p, q)^T)$
      Izračunaj $(\tilde{u_j}, \tilde{v_j})$, t_j, e_j
    kraj
    Sortiraj t_j, $j = 1, \ldots, m$ i e_j, $j = 1, \ldots, m$ po kriteriju $|e_j| \leq |e_{j+1}|$
    $\hat{k} \leftarrow 0$
    za $k \leftarrow k_{min}$ **do** m **čini**
      Izračunaj p_k^{KS}, p_k^{JB}, p_k^t na osnovu t_j i e_j, $j = 1, \ldots, k$
      ako $\min\{p_k^{KS}, p_k^{JB}, p_k^t\} > p_{min}$ **onda**
        $\hat{k} \leftarrow k$
      kraj
    kraj
    $\pi^{(1)} \leftarrow \{a_j \in \mathcal{S} : j = 1, \ldots, \hat{k}\}$
    ako $|\pi^{(1)}| > |\pi^{(0)}|$ **onda**
      $\pi^{(0)} \leftarrow \pi^{(1)}$
      $E^{(0)} \leftarrow E^{(1)}$
    kraj
  kraj
kraj
vrati $\pi^{(0)}$, $E^{(0)}$

Algoritam 1: Prepoznavanje elipsi

Neka su $(p_s, q_s, \theta_s, a_s, b_s)$ parametri elipse pronađene u iteraciji s. Kako bi olakšali računanje, translatiramo i rotiramo svih m točaka skupa $\mathcal{A}$

$$(u_j, v_j)^T = U^T((x_j, y_j)^T - (p_s, q_s)^T),$$

[1]Random Sample Consensus, može se interpretirati kao iterativna metoda za prepoznavanje stršećih vrijednosti skupa podataka

gdje je U dana s (4). Na ovaj način razmatranje svodimo na elipsu $(0, 0, 0, a_s, b_s)$. U sljedećem koraku, računamo projekciju $(\tilde{u_j}, \tilde{v_j})$ svake točke (u_j, v_j), $j = 1, \ldots, m$ na elipsu. Ova projekcija ima svojstvo da normala na elipsu u točki $(\tilde{u_j}, \tilde{v_j})$ prolazi kroz točku (u_j, v_j). Za više detalja pogledati [3, Poglavlje 14.13.1]. Koristeći parametrizaciju (1) svaka točka $(\tilde{u_j}, \tilde{v_j})$ na elipsi može se prikazati kao $(x(t_j), y(t_j))$ za neki $t_j \in [0, 2\pi)$. Nadalje, označimo sa e_j Euklidsku udaljenost točki (u_j, v_j) i $(\tilde{u_j}, \tilde{v_j})$, koja ima negativan predznak za točke unutar elipse, a pozitivan inače.

Kako bi ispitali dolaze li podaci sa elipse u trenutnoj iteraciji, koristeći model elipse dan u [4], potrebno je primjeniti sljedeće statističke testove:

- Kolomogorov-Smirnov test

  Ovaj test ispituje jesu li točke, predstavljene kutevima t_j, $j = 1, \ldots, k$, uniformno raspoređene po obodu elipse. Drugim riječima, zahtijeva se da je vjerojatnost odabiranja točke na nekom luku elipse razmjerna duljini luka.

  Primjedba. *Originalni algoritam koristi Kuipersov test. Prednost potonjeg jest da ne umanjuje važnost razlika u repovima distribucija, što je prikladno za ovu primjenu, jer se radi o odstupanjima po elipsi.*

- Jarque-Bera test

  Testira dolaze li udaljenosti e_j, $j = 1, \ldots, k$ iz normalne distribucije.

- t-test

  Provjerava je li očekivanje distribucije udaljenosti e_j, $j = 1, \ldots, k$ jednako nuli.

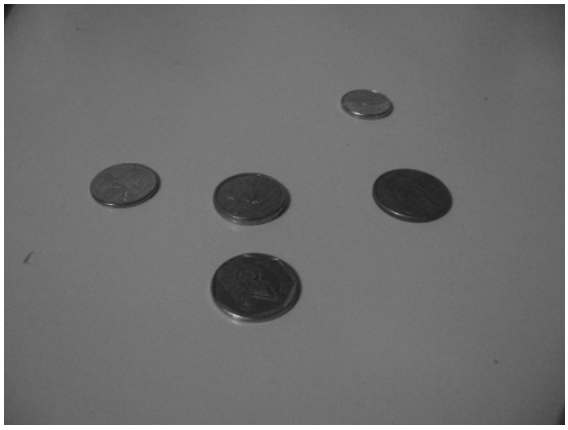
Svaki od navedenih testova kao rezultat daje p-vrijednost, koja nam govori o vjerojatnosti da je rezultat točan, tj. da nije točan. Ukoliko su sve p-vrijednosti veće od p_{min}, zaključujemo da podaci e_j, $j = 1, \ldots, k$ i t_j, $j = 1, \ldots, k$ dolaze sa elipse u trenutnoj iteraciji. Često se uzima $p_{min} = 0.05$.

Kako su podaci sortirani uzlaznim redoslijedom prema Euklidskim udaljenostima do elipse, prirodno je pretpostaviti da su prvih k podataka najbliži elipsi. Na liniji 13, iterirajući, pokušavamo što više točaka smjestiti na elipsu, tj. biramo $\hat{k} = \max\{k : \min\{p_k^{KS}, p_k^{JB}, p_k^t\} > p_{min}\}$. Kako bi uzeli dovoljno točaka u obzir, k_{min} bi trebao biti veći od 20. Na ovaj način znamo koliko točaka pripada elipsi u trenutnoj iteraciji. Kroz N pokušaja, najbolja elipsa se bira tako da je $\hat{k}$ najveći mogući. Na izlazu dobivamo elipsu $E^{(0)}$ i skup točaka koji joj pripada $\pi^{(0)}$.

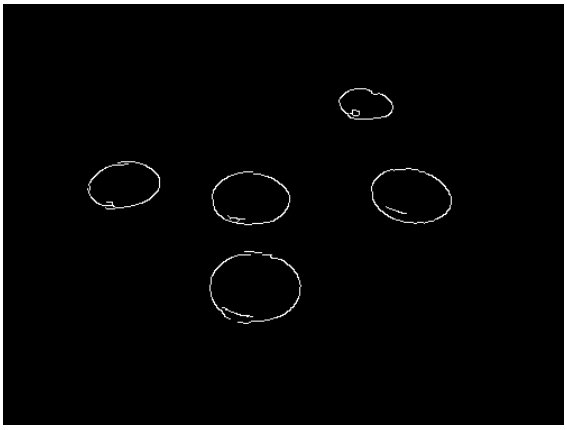
Kako je provođenje velikog broja statističkih testova vremenski zahtijevno, potrebno ih je aproksimirati. U praktičnom dijelu završnog izrađen je program koji se temelji na algoritmu 1 u programskom jeziku C++. Detalji implementacije izneseni su u sljedećem poglavlju.

6 Implementacija algoritma

Za implementaciju algoritma iz prethodnog poglavlja korištena je OpenCV[2] bibilioteka ([2]). Algoritam na ulazu dobiva proizvoljnu sliku. Potrebno je najprije detektirati rubove, kako bi od realne slike načinili digitalnu sliku. U programu se poziva funkcija `cv::Canny` koja detektira rubove pomoću Canny filtera. Djelovanje Canny filtera možemo vidjeti na slikama 4 i 5.



Slika 4: Kovanice



Slika 5: Rubovi

Iterirajući kroz piksele slike dobivene Canny filterom, spremimo koordinate točki rubova na kojima će algoritam raditi. Prethodno je implementirano u funkciji `getPoints`.

Na linijama 4 do 8 postavljaju se vrijednosti parametara za algoritam, definiraju se `ellSol`, u kojem će biti spremljena rješenja, i `vecInd`, polje nula i jedinica u kojem nula označava da je podatak slobodan, a jedan da već pripada nekoj elipsi. Algoritam se izvršava sve dok ima barem `minPts` slobodnih točaka. U tom slučaju obavlja se `N` pokušaja. Elipsa koja najbolje predstavlja slobodne točke sprema se u `bestEll`, dok se pripadni indeksi točaka spremaju u `delInd`. Navedeni indeksi koriste se u 27 i 28 pri ažuriranju polja `vecInd`. Broj točaka koje je elipsa obuhvatila spremljen je u `nrPtsEll`. Moguće je i postaviti donju među na broj točaka koje elipsa mora obuhvatiti. Jedan takav zaustavni uvjet je na liniji 21.

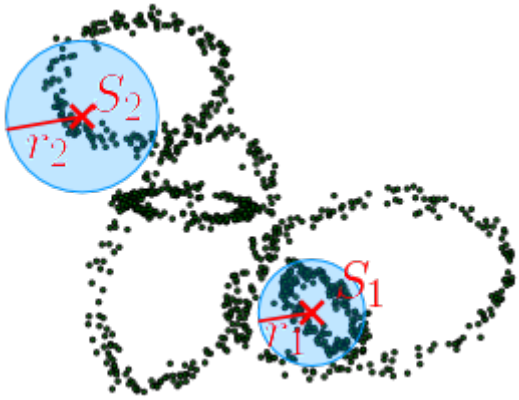
```
1  void algorithm(cv::Mat image)
2  {
3      std::vector<std::pair<int,int>> A = getPoints(image);
4      int N = 1000, minPts = 50, thr = 0.085;
5      int m = A.size()
6      int freePts = m;
7      std::vector<std::vector<double>> ellSol;
8      std::vector<bool> vecInd(m);
9      while (freePts > minPts)
10     {
11         std::vector<double> bestEll;
12         std::vector<int> delInd;
13         int nrPtsEll = 0;
14         double scaledNrPts = 0;
15         int br = 0;
16         while (br < N)
17         {
18             //...
19         }
```

[2]Open Source Computer Vision Library

```
20
21          if (nrPtsEll < thr * m)
22              break;
23
24          ellSol.push_back(bestEll);
25          freePts -= nrPtsEll;
26
27          for (int d : delInd)
28              vecInd[d] = true;
29      }
30  }
```

Preostalo je analizirati što se doagađa na svakoj od `N` iteracija. Najprije funkcija `pick`, za dani skup slobodnih točaka iz `A`, bira m_e točaka koje će određivati elipsu. Najmanji mogući odabir za m_e je 5, što ova implementacija koristi. Naime, potrebno je pet točaka, među kojima nikoje tri nisu kolinearne, kako bi konika bila jednoznačno određena.

Primjedba. *Nije u potpunosti jasno kako funkcija* `pick` *bira točke iz A. Naime, može se dogoditi da na svakoj iteraciji odabere 5 točaka koje dolaze sa različitih elipsi, pa nema uopće smisla ocijenjivati elipse tj. tražiti najbolju među njima. Kako bi kroz* `N` *iteracija dobili što više reprezentativnih elipsi, jedna ideja je točke birati lokalizirano. Odabere se slučajna točka S_i i koristiti kao sidro. U krugu radijusa r_i sa središtem u S_i biraju se slučajnih m_e točaka. Kroz iteracije se r_i može mijenjati, kako bi prepoznali elipse svih veličina. Ilustracija ove primjedbe je na slici 6.*

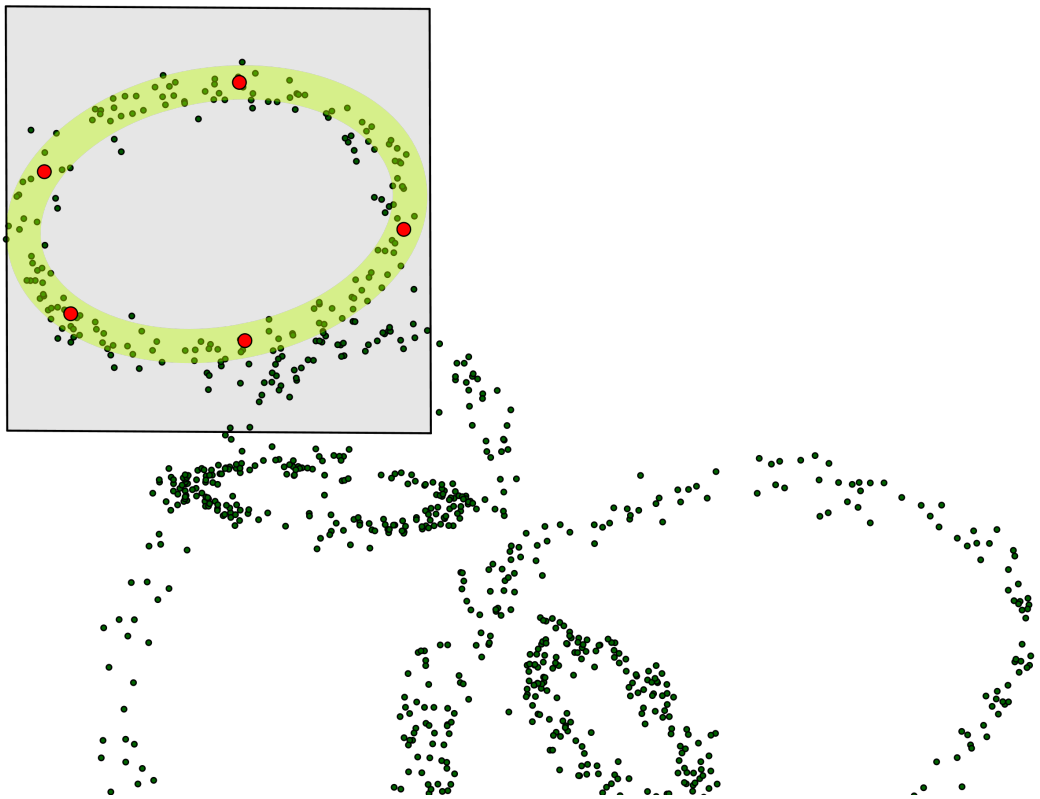


Slika 6: Lokalizirani odabir točaka

U funkciji `findConicSVD` izračunaju se koeficijenti A, B, C, D, E i F konike određene s m_e odabranih točaka. Za pronađene koeficijente, funkcija `identifyConic` provjerava je li dobivena konika elipsa i koristeći znanja iz poglavlja 2, odredi obilježja elipse (p, q, θ, a, b), tj. središte elipse, kut do standardne orijentacije te duljine poluosi.

U slučaju da imamo elipsu, umjesto projiciranja svih točaka, na elipsu projiciramo samo točke u kvadratu stranice duljine `2(a + pixRange)`. Provjeru pripadnosti točke unutrašnjosti spomenutog kvadrata (slika 7) vrši funkcija `inSquare`, koja vraća `true` ukoliko je točka u spomenutome kvadratu, a `false` inače.

Ne provode se brojni statistički testovi, već se isti aproksimiraju. Područje oko elipse u pojasu širine `pixRange` (slika 7) podijelimo na koševe (`bins`), tako da su duljine lukova koje pripadaju pojedinom košu jednake. Sve točke u spomenutom pojasu se projiciraju na elipsu funkcijom `projectionEll`, koja vraća kut



Slika 7: Kvadrat i pojas oko elipse

`t` iz parametrizacije elipse i Euklidsku udaljenost `e` kao što je opisano u prethodnom poglavlju. Ove podatke koristimo kako bi ispitali pripadnost točke pojasu oko elipse i pronalaženju koša kojemu točka pripada.

Na liniji 34 je kriteriji za ocjenu elipsi. Ta ocjena ovisi o:

a. ukupnom broju točaka u pojasu oko elipse

b. broju točaka koje treba premjestiti kako bi dobili uniformnu raspodjelu točaka po koševima na elipsi

Funkcija `findDev` vraća broj iz $[0, 1]$, koji je omjer brojeva iz a i b.

U varijabli `sumPts` će se nalaziti broj točaka u pojasu širine `pixRange` oko elipse u trenutnoj iteraciji, dok će se indeksi točaka u tom pojasu spremati u `ind`.

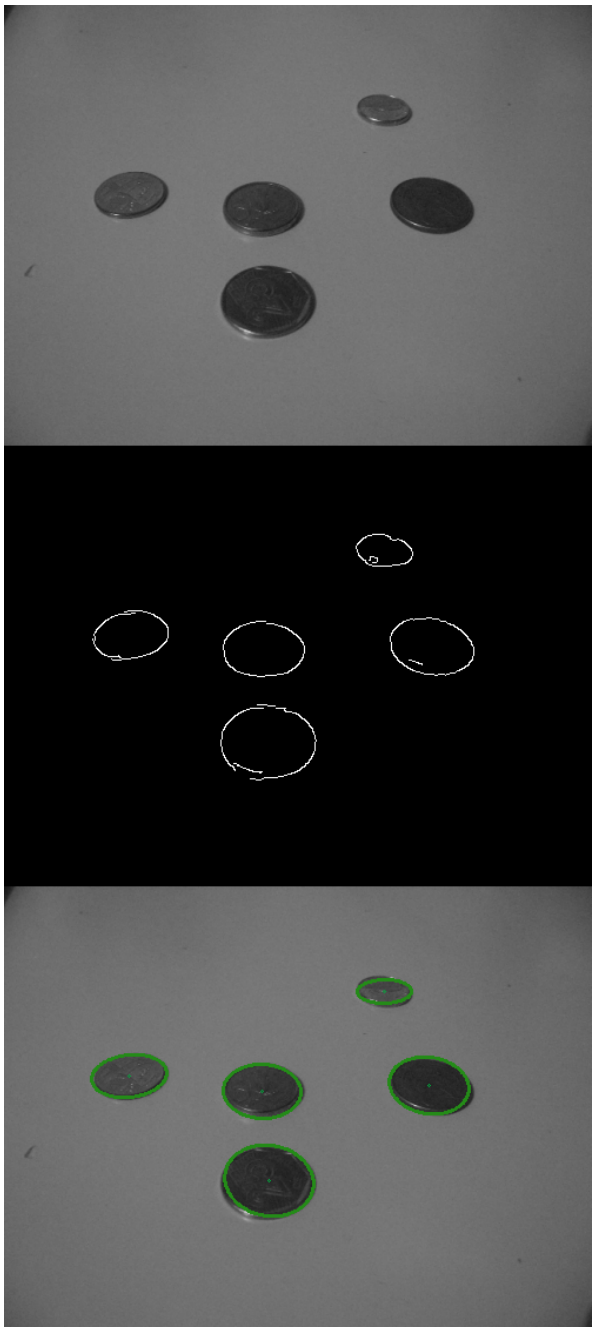
```
1  enum ellData {p, q, theta, a, b};
2  //...
3  while (br < N)
4  {
5      std::vector<std::pair<int,int>> el = pick(A, vecInd);
6      std::vector<double> C = findConicSVD(el);
7      std::vector<double> D = identifyConic(C);
8      int nrBins = 30, pixRange = 3;
9
10     if (isEllipse(D))
11     {
12         ++br;
13         int sumPts = 0;
14         std::vector<int> bins(nrBins);
15         std::vector<int> ind;
16
```

```
17          for (int i = 0; i < m; ++i)
18          {
19              if (!vecInd[i] && inSquare(A[i], D))
20              {
21                  double t, e;
22                  std::tie(t, e) = projectionEll(A[i], D);
23                  if (std::abs(e) < pixRange)
24                  {
25                      bins[findBin(t)]++;
26                      sumPts++;
27                      ind.push_back(i);
28                  }
29              }
30          }
31
32          double deviation = findDev(bins);
33
34          if ((1 - deviation) * sumPts > scaledNrPts)
35          {
36              scaledNrPts = (1 - deviation) * sumPts;
37              nrPtsEll = sumPts;
38              bestEll = D;
39              delInd = ind;
40          }
41      }
42  }
```

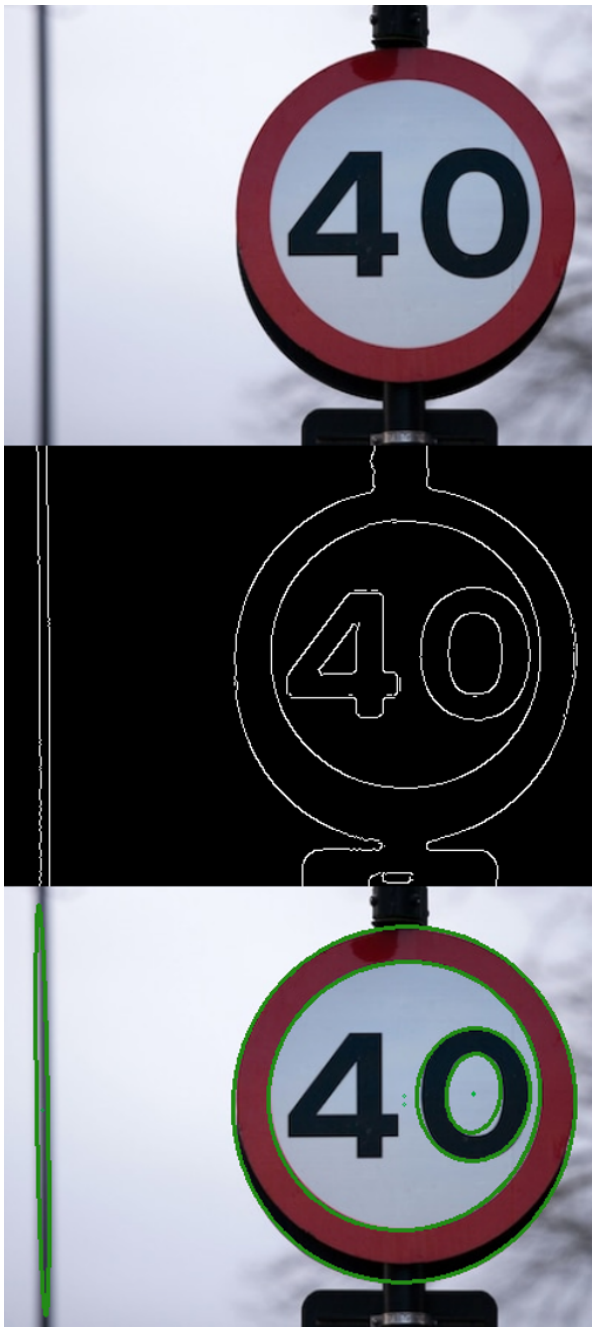
7 Numerički eksperimenti

Implementacija algoritma iz prethodnog poglavlja testirana je na mnoštvu primjera. Neki od njih imaju jasne rubove, dok neki imaju diskretne rubove. Na vrhu slika 8 i 9 su prikazane kovanice i prometni znak. Odmah ispod njih su rubovi koje je prepoznao Canny filter. Na dnu slika su elipse koje je program pronašao. Treba napomenuti da rad programa uvelike ovisi o filteru rubova. Program se može testirati na `http://cs.mathos.unios.hr/~pnikic/web.php`.

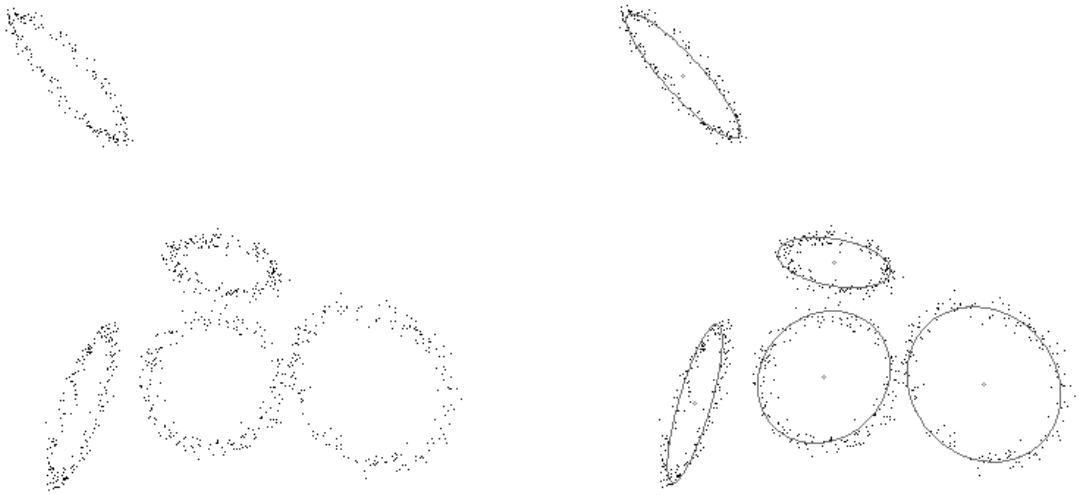
Program je testiran i na 100 generiranih primjera, od kojih 50 sadržavaju razdvojene, a ostalih 50 preklapajuće elipse. Pripadni podaci $\mathcal{A}_1 - \mathcal{A}_{100}$ dolaze svaki od 5 elipsi. Algoritam ima vrlo visoku stopu prepoznavanja elipsi. Rezultati nekih testiranja dani su na slikama 10, 11 i 12.



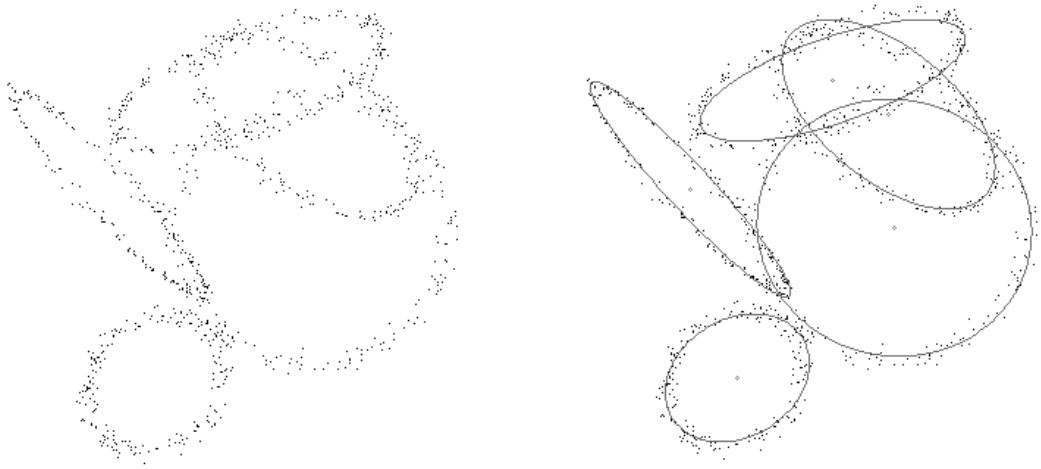
Slika 8: Eksperiment 1



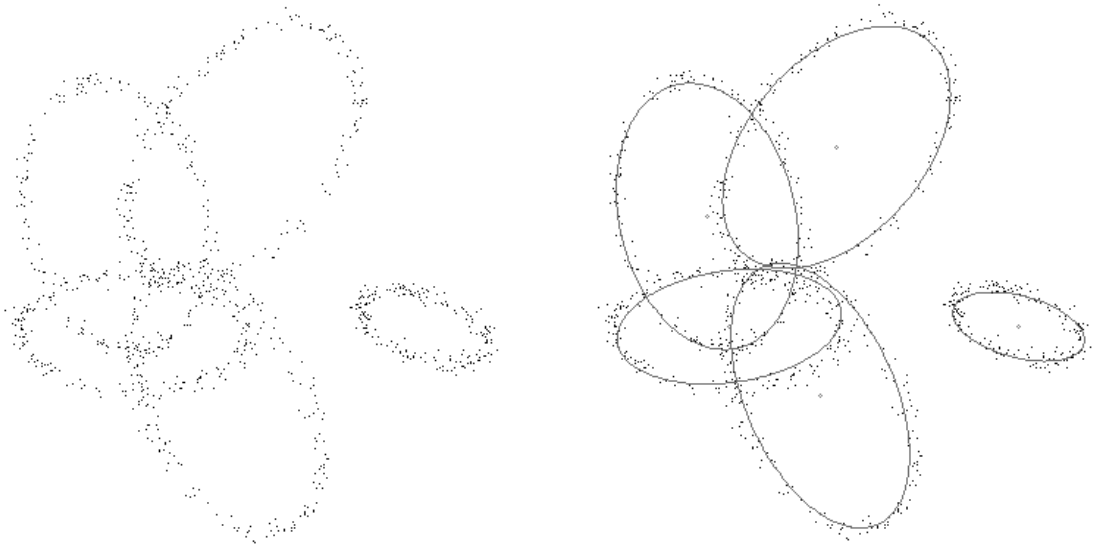
Slika 9: Eksperiment 2



Slika 10: Eksperiment 3



Slika 11: Eksperiment 4



Slika 12: Eksperiment 5

Literatura

[1] C. Akinlar, C. Topal, EDCircles: A real-time circle detector with a false detection control, Pattern Recognition 46(2013), 725-740.

[2] G. Bradski, The OpenCV Library, Dr. Dobb's Journal of Software Tools, 2000.

[3] D.H. Eberly, 3D game engine design, Morgan Kaufmann Publishers, San Francisco, 2006.

[4] R. Grbić, D.Grahovac, R. Scitovski, A method for solving the multiple ellipses detection problem, Pattern Recognition 60(2016), 824-834.

[5] Z.-Y. Liu, H.Qiao, Multiple ellipses detection in noisy environments: a hierarchical approach, Pattern Recognition 42(2009), 2421-2433.

[6] D.K. Prasad, M.K.H. Leung, C.Quek, Ellifit: an unconstrained, non-iterative, least squares based geometric ellipse fitting method, Pattern Recognition 46(2013) 1449-1465.

[7] R. Scitovski, T. Marošević, Multiple circle detection based on center-based clustering, Pattern Recognition Letters 52(2014), 9-16.