Poročilo: Magnetno polje Helmholtzove tuljave (23. naloga)

Anton Luka Šijanec, 3. a

V petek, 3. junija 2022

Povzetek

Poročilo merjenja magnetnega polja Helmholtzove tuljave vzdolž dveh geometrijskih osi in primerjave rezultatov s teoretično napovedjo pri fiziki na Gimnaziji Bežigrad.

Kazalo

Opomba Uvod in teorija, z izjemo numerične obdelave, sta enaka kot v opisu projekta.

1 Uvod

Helmholtzova tuljava oziroma Helmholtzov par je poimenovana po nemškemu fiziku devetnajstega stoletja Hermannu Ludwigu Ferdinandu von Helmholtzu in predstavlja pripravo za izdelavo umetnega lahko dostopnega homogenega magnetnega polja visoke kvalitete. Solenoidi, torej dolge in ozke tuljave, pod električnim poljem sicer delajo kvalitetno, z vstavljenim feromagnetom tudi močno magnetno polje, vendar je le-to težko dostopno, saj se nahaja znotraj navojev. Helmholtzova tuljava pa je sestavljena iz dveh tokovnih zank oziroma magnetnih dipolov, to sta dve ravninsko vzporedni enaki kratki in široki med seboj za polmer oddaljeni tuljavi. Ob prisotnosti enako velikega in enako usmerjenega električnega toka v vodnikih se v sredini med njima pojavi veliko območje s homogenim magnetnim poljem, kar je uporabno za mnoge elektrotehnične probleme, denimo natančno kalibracijo merilnikov in nasprotovanje vplivu Zemljinega magnetnega polja na instrumente.

Maxwellova tuljava škotskega fizika Jamesa Clerka Maxwella je dodatek Helmholtzovi tuljavi. Z dodatnimi pravilno postavljenimi navitji za ceno dodatnega materiala in kompleksnosti dodatno poveča območje in homogenost magnetnega polja.

2 Teorija

Navodilo zahteva meritev in teoretično napoved vrednosti magnetnega polja na oseh tuljave. Če je radij R, število navojev n in tok I, je jakost magnetnega polja B na sredini

$$B = \left(\frac{4}{5}\right)^{3/2} \frac{\mu_0 n I}{R},$$

kjer je μ_0 indukcijska/magnetna konstanta. ($\pi 4\text{e}{-7}\,\mathrm{V\,s\,A^{-1}\,m^{-1}}$).

2.1 Razlaga enačbe

2.1.1 Biot-Savartov zakon

Z Biot-Savartovim zakonom predstavimo vektor B magnetnega polja, ki se pojavi zaradi stalnega električnega polja I v vodniku na poti C. Predstavi relacijo magnetnega polja na jakost, smer, dolžino in bližino električnega toka z enačbo

$$B(r) = \frac{\mu_0}{4\pi} \oint_C \frac{I d\ell \times \hat{r}'}{|r'|^2},$$

kjer je $d\ell$ vektor na poti elektronov C (krožnica), katerega velikost je infinitezimalno majhen del vodnika v smeri električnega toka, ℓ je torej točka na C, $\hat{r}'$ je enotski vektor (množen z obratno vrednostjo svoje velikosti — $\hat{r}' = r' \cdot |r'|^{-1}$) vektorja $r' = r - \ell$ — vektorja premika od vodnika ($d\ell$ oz. ℓ) do točke r, katere vektor magnetnega polja želimo izračunati. Od prej je μ_0 še vedno indukcijska/magnetna konstanta.

Čeprav se zakon povečini, kot v našem primeru, uporablja na zankah, v katerih teče električni tok, velja tudi za neskončno dolge vodnike. Na tak način je bil z Biot-Savartovim in Lorentzovim zakonom na primer do leta 2019 SI amper definiran kot konstantani tok, ki v vakuumu, ko teče po dveh eden od drugega za en meter oddaljenih vzporednih vodnikih neskončne dolžine z zanemarljivim premerom, povzroči nastanek sile $2\text{e}{-7}\,\mathrm{N}$ na vsak meter dolžine vodnika.

2.2 Polje na osi zanke skozi obe navitji

Biot-Savartov zakon poenostavimo in izračunamo vektorja B za na osi zanke s polmerom R vodnika, po katerem teče tok I, poljubno točko T, oddaljeno z od središča zanke S.

Zanima nas samo z osjo vzporedna komponenta B_z vektorja magnetnega polja, ker na os pravokotne komponente v homogenem polju sploh ne bo, ko bomo računali jakost polja znotraj Helmholtzove tuljave.

Kot med na os pravokotno in z osjo vzporedno komponento vektorja B je enak kotu (ℓ, T, S). Torej je

$$\sin\theta = \frac{R}{r = \sqrt{z^2 + R^2}} \wedge dB_Z = dB \sin\theta \Longrightarrow dB_Z = \frac{\mu_0 I d\ell}{4\pi} \cdot \frac{R}{\left(z^2 + R^2\right)^{3/2}}.$$

Vse vrednosti, razen infinitezimale dolžine na zanki $d\ell$, so konstantne, integriranje $d\ell$ pa predstavlja obseg zanke $2\pi R$. Tako je velikost z osjo zanke vzporedne komponente vektorja magnetenega polja v točki T izračunana z enačbo

$$B_Z = \frac{\mu_0 I}{\cancel{4\pi}^{\,2}} \cdot \frac{\cancel{2\pi} R^2}{\left(z^2 + R^2\right)^{3/2}}.$$

Ker so vse točke $d\ell$ na zanki enako oddaljene od točke T, je za točke z na osi tuljav $B_Z(z) = B(z)$.

2.2.1 Dodajanje zank

Da dobimo enačbo za polje enega navitja Helmholtzove tuljave, enačbo za eno zanko preprosto zmnožimo s številom zank n:

$$B_1(z) = \frac{n\mu_0 R^2 I}{2\left(z^2 + R^2\right)^{3/2}}.$$

Zanima nas taka vrednost z, ki je v središču med obema navitjema. To je, kot je zgoraj opisano, $z = R/2$:

$$B_1\left(\frac{R}{2}\right) = \frac{n\mu_0 R^2 I}{2\left(\left(\frac{R}{2}\right)^2 + R^2\right)^{3/2}}.$$

Jakost magnetnega polja med obema simetričnima navitjema je enaka dvakratniku B_1.

$$B\left(\frac{R}{2}\right) = 2B_{1\,Z}\left(\frac{R}{2}\right) =$$

$$= \frac{\cancel{2}n\mu_0 R^2 I}{\cancel{2}\left(\left(\frac{R}{2}\right)^2 + R^2\right)^{3/2}} = \frac{n\mu_0 R^2 I}{\left(\frac{1}{4}R^2 + R^2\right)^{3/2}} = \frac{n\mu_0 R^2 I}{\left(\frac{5}{4}R^2\right)^{3/2}} = \left(\frac{4}{5}\right)^{3/2} \frac{n\mu_0 \cancel{R^2} I}{\cancel{R^3}} = \left(\frac{8}{5\sqrt{5}}\right) \frac{n\mu I}{R}.$$

2.3 Potreben tok za dosego željenega homogenega toka

Zgornjo enačbo preuredimo v

$$I = \left(\frac{5}{4}\right)^{3/2} \left(\frac{BR}{\mu_0 n}\right),$$

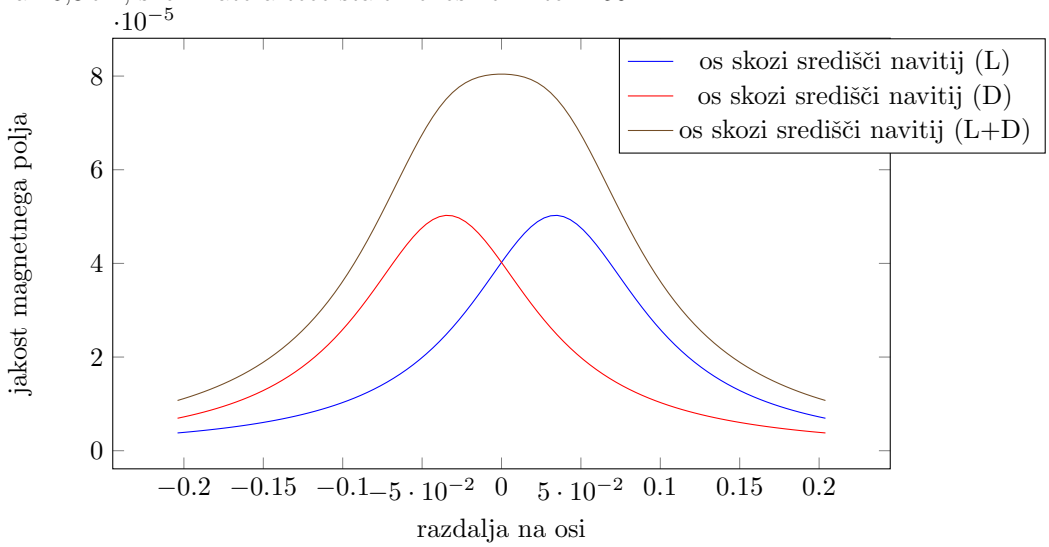
kjer je μ_0 malo drugače zapisana ista indukcijska/magnetna konstanta ($= \pi 4\mathrm{e}{-7}\,\mathrm{T\,m\,A^{-1}}$), in s tem izračunamo potreben tok za stalno homogeno magnetno polje v središču tuljave.

2.4 Polje na simetrijski osi

Druge osi nisem znal izračunati, zato bom teoretično napoved izračunal numerično - predstavljal si bom, da je namesto okrogle zanke tam mnogokotnik in uporabil Biot-Savartov zakon v iteraciji z računalnikom.

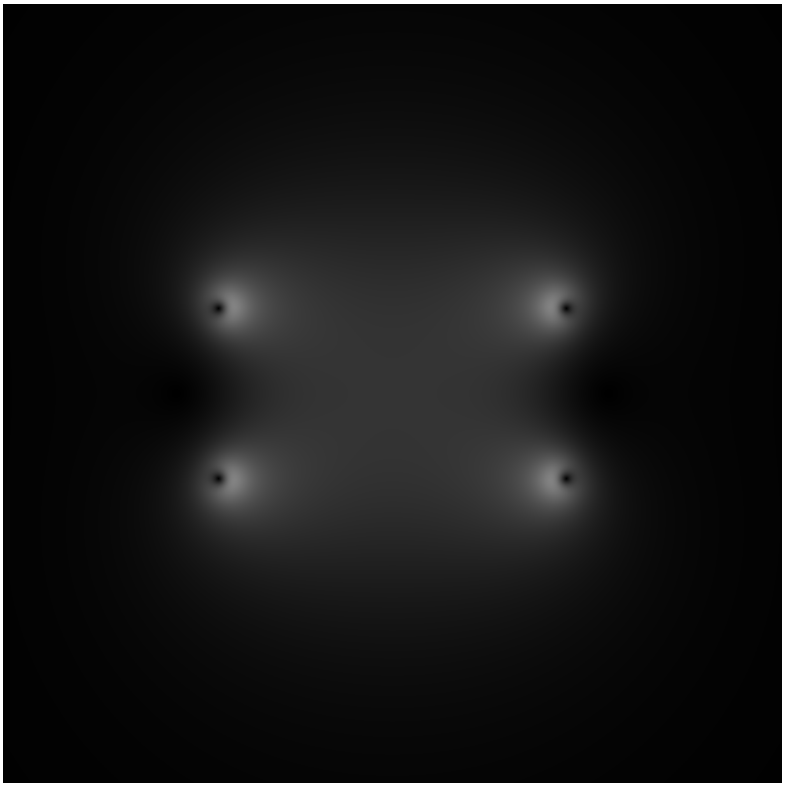
2.5 Grafi

Kot primer vzemimo šolsko Helmholzovo tuljavo s 320 navoji na vsakem navitju, katerega radij je enak 6,8 cm, skozi katera teče stalen enosmerni tok 250 mA.



2.6 Numerična analiza

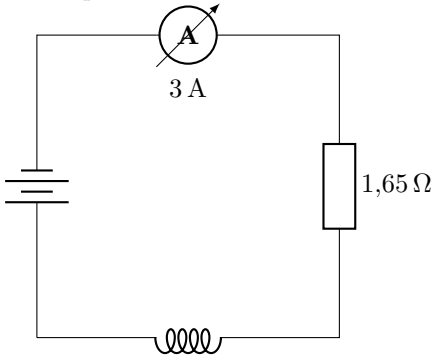
Numerična analiza je preprosta; namesto zanke v obliki kroga seštejemo vrednosti na odsekih vodnika, ki predstavlja mnogokotnik v 3D prostoru. Program za numerično analizo je priložen dokumentu in, v kolikor generira za mojo eksperimentalno tuljavo (`./numerično.c 0.088 3 17 0.0002 1000 30 pgm > izhod.pgm`), izdela sledeč graf:



Slika 1: Teoretični model tuljave

3 Praktične meritve

Kot je napisano v opisu projekta, sem uporabil TDKjev MPU9250 merilnik in mikrokrmilnik Espressif 8266, ki je vsakih nekaj milisekund izmeril absolutno jakost magnetnega polja. Za določanje pozicije merilnika ob danem času sem uporabil funkcijo *Motion Tracking* v programu Blender. Korelacija podatkov iz dveh virov s stališča fizike ni pomembna, vendar je vseeno bežno opisana v predstavitvi.



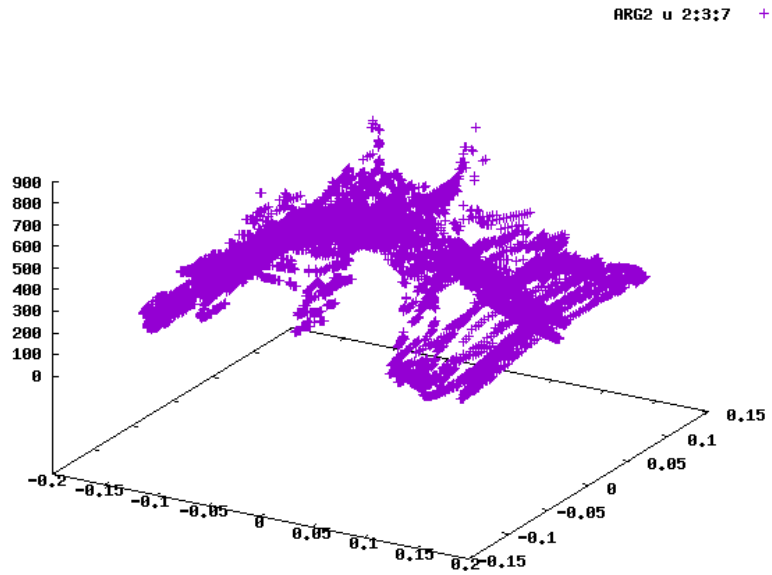
3.1 Pripomočki

- 3 A 5 V napajalnik
- štirje 10 W 2,2 Ω uporniki za nadomestno upornost vsaj 1,65 Ω in zmožnost porabe 20 W – v resnici je bila teoretična poraba 15 V A
- lakirana bakrena žica za navitji s polmerom 8,8 cm, pridobljena iz toroidnega transformatorja
- kamera prenosnega telefona za videoanalizo
- Hallov merilnik mangnetnega polja

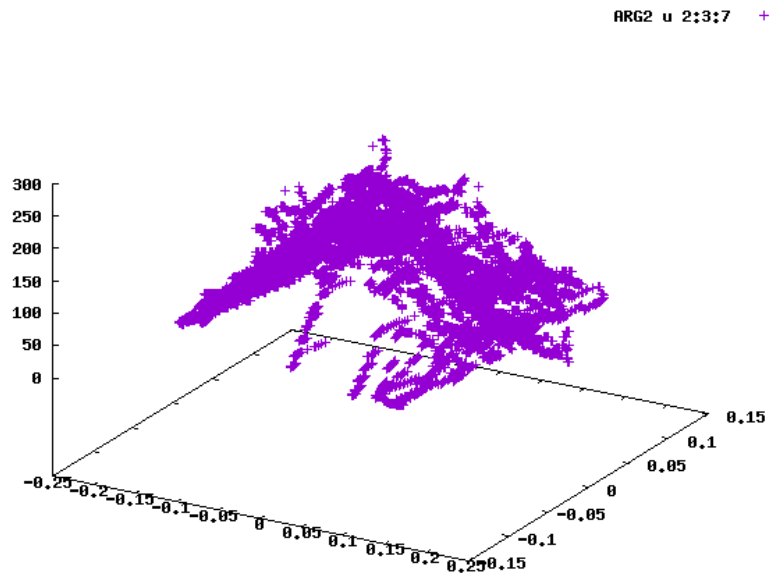
3.2 Opis rezultatov

Dobil sem tabelo s 24553 vrsticami. Opravil sem osem uspelih meritev, od tega tri na troamperskem toku, pet pa na dvoamperskem toku. Tabeli meritev sta dostopni na `http://git.sijanec.eu/sijanec/sola-gimb-3/src/branch/master/fiz/naloga/relacije`.

3.3 Slike grafov rezultatov



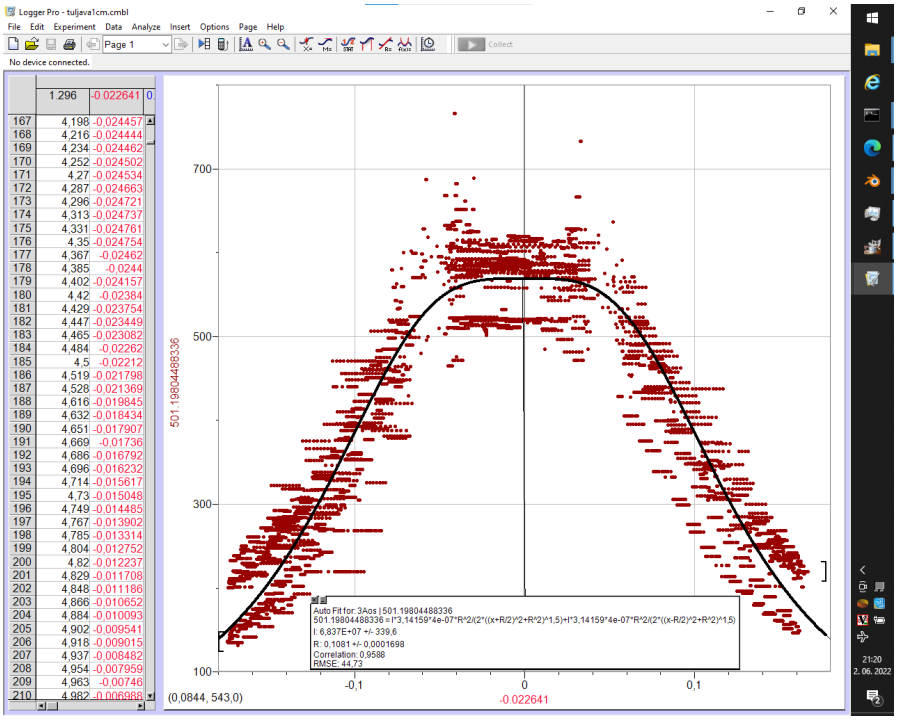
Slika 2: Graf magnetnega polja v odvisnosti od pozicije pri triamperskem toku



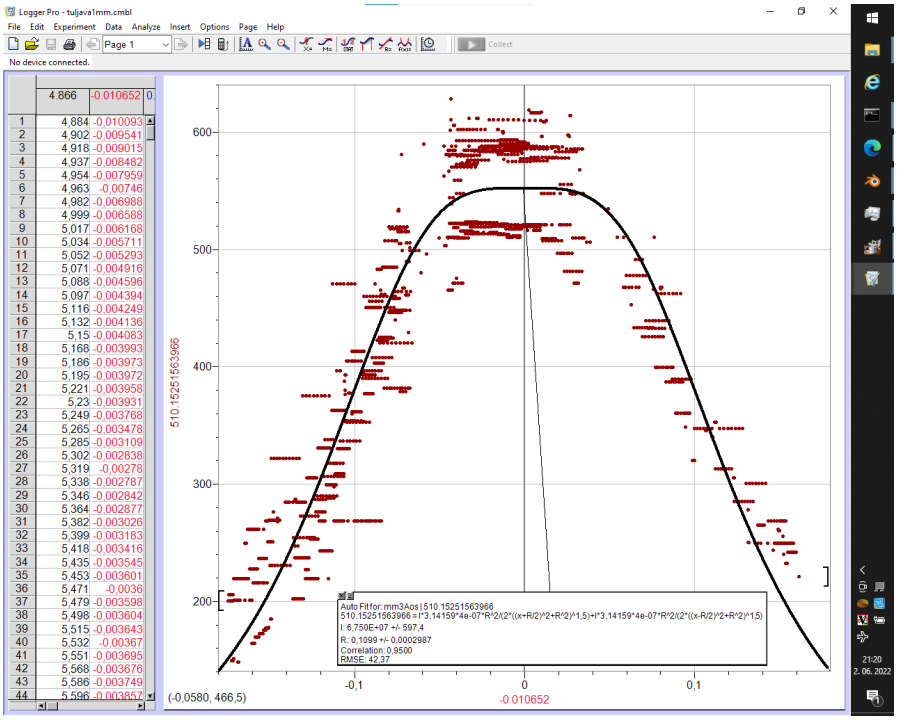
Slika 3: Graf magnetnega polja v odvisnosti od pozicije pri dvoamperskem toku

3.4 Primerjava s teoretično napovedjo

Podatke sem uvozil v LoggerPro in izdelal prilagoditveno krivuljo. Iz meritev sem izluščil najprej centimeterski in nato še milimetrski pas okoli osi.



Slika 4: Grafa meritev in prilagoditvene krivulje, če je pas centimetrski



Slika 5: Grafa meritev in prilagoditvene krivulje, če je pas milimetrski

4 Viri

Internetni viri

[B.d.]. **urlalso**: http://hyperphysics.phy-astr.gsu.edu/hbase/magnetic/curloo.html#c4.

[B.d.]. **urlalso**: http://hyperphysics.phy-astr.gsu.edu/hbase/magnetic/curloo.html#c3.

[B.d.]. **urlalso**: http://sl.wikipedia.org/wiki/Hallov_pojav.

[B.d.]. **urlalso**: http://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf.

[B.d.]. **urlalso**: http://en.wikipedia.org/wiki/Helmholtz_coil.

[B.d.]. **urlalso**: http://lbm.fe.uni-lj.si/oe/OE2/LabVaja/Priprave%20za%201VAJO%200E2%20V2.docx.

[B.d.]. **urlalso**: http://en.wikipedia.org/wiki/Biot%E2%80%93Savart_law.

[B.d.]. **urlalso**: http://sl.wikipedia.org/wiki/Infinitezimala.

[B.d.]. **urlalso**: http://en.wikipedia.org/wiki/Lorentz_force.

[B.d.]. **urlalso**: http://hyperphysics.phy-astr.gsu.edu/hbase/magnetic/Biosav.html#c1.

[B.d.]. **urlalso**: http://en.wikipedia.org/wiki/Ampere#Former_definition_in_the_SI.

[B.d.]. **urlalso**: http://en.wikipedia.org/wiki/Rotation_matrix.

5 Zaključek

Prilagoditvena krivulja v programu LoggerPro je ustrezno natančna.

6 Priloge

6.1 Program za teoretičen izris magnetnega polja okoli tuljave

Animaciji dveh tuljav, ki se premikata po prostoru: `http://video.arnes.si/?q=helmholtz`. Ena je izdelana s hitrejšim programom (in večjim, da ga res nima smisla vključevati v dokument) v OpenGL, druga pa s tem programom.

```
#include <stdlib.h>
#include <stdio.h>
#include <error.h>
#include <math.h>
#include <signal.h>
#include <string.h>
#include <errno.h>
#define UVOD "program za numeri ni izra un jakosti magnetnega polja okoli helmholtzove tuljave\n" \
  "sem spisal anton luka ijanec za projektno nalogo pri fiziki v tretjem letniku gimb.\n" \
  "uporaba: %s in nujni argumenti po vrsti:\n" \
  " 1. radij enega navitja v metrih\n" \
  " 2. tok, ki te e po vodniku v amperih\n" \
  " 3. tevilo navojev na enem navitju\n" \
  " 4. razmak med merilnimi to kami v metrih\n" \
  " 5. koliko meritev od sredi a v obe dimenziji naj napravimo\n" \
  " 6. koliko kotov naj ima navitje - ra unamo, kot da je mnogokotnik\n" \
  "nenujni argumenti: prazen niz argument nastavi na privzeto vrednost.\n" \
  " 7. tip izhodnih podatkov (pgm, ppm ali tsv). privzeto je to pgm.\n" \
  " 8. razmak med navitjima. privzeto je to r/2.\n" \
  " 9. zamik enega navitja v metrih. privzeto sta osi navitij ista premica.\n" \
  " A. polmer druge tuljave. privzeto je enak polmeru prve tuljave.\n" \
  " B. tok druge tuljave v metrih. privzeto je enak toku prve tuljave.\n" \
  "oblika izhodnih podatkov, e je 7. parameter pgm, so pgm slike z vrednostmi 0-255\n" \
  " - slika je prerez tuljave. magnetno polje te e vodoravno.\n" \
  " - vrednosti direktno korelirajo z izra unano jakostjo v decigaussih: 10e-5 tesla\n" \
  " - slika je iroka 1+2*koliko in visoka 1+2*koliko (5. argument) slikovnih to k\n" \
  "oblika izhodnih podatkov, e je 7. parameter ppm, so barvne slike z 8 biti na barvo\n" \
  " - barvi rde a in zelena predstavljata komponenti vektorjev polja i in j\n" \
  "oblika izhodnih podatkov, e je 7. parameter tsv, je tsv, z naslednjimi stolpci:\n" \
  " 1. komponenta i krajevnega vektorja to ke meritve. 0,0 je v sredi u tuljave.\n" \
  " 2. komponenta j krajevnega vektorja to ke meritve. 0,0 je v sredi u tuljave.\n" \
  " 3. komponenta i vektorja magnetnega polja v to ki meritve.\n" \
  " 4. komponenta j vektorja magnetnega polja v to ki meritve.\n" \
  " 5. jakost magnetnega polja v teslah - tokrat ni v decigaussih!\n" \
  "na in izdelave animacije: podan naj bo samo en argument - animacija - TOLE NE DELA\n" \
  " - delajo se datoteke animacija0000.ppm, animacija0001.ppm, ...\n" \
  " - parametri se sinusoidno spreminjajo po vdelanih konstantah.\n" \
```

```
38    " - slike lahko recimo s ffmpeg(1) nato pretvorite v videoposnetek\n" \
39    "DODATEK :: izra un za eno to ko:\n" \
40    " - 1. argument je 'enkrat', 2. R, 3. x, 4. y, 5. kotov \n" \
41    " - x in y koordinati sta v metrih. program izpi e eno vrstico\n" \
42    " - v vrstici so i, j in k komponente polja in absolutna vrednost.\n"
43  enum oblika {
44    PGM,
45    PPM,
46    TSV,
47    ANIMACIJA
48  };
49  struct vektor {
50    long double i; // x - desno na sliki
51    long double j; // y - gor na sliki
52    long double k; // z - v monitor
53  };
54  struct vektor se tej (struct vektor a, struct vektor b) {
55    struct vektor r = {
56      .i = a.i + b.i,
57      .j = a.j + b.j,
58      .k = a.k + b.k,
59    };
60    return r;
61  }
62  struct vektor vektorski_produkt (struct vektor a, struct vektor b) {  // ne bom
        implementiral
63    struct vektor r = {           // matrik
64      .i = a.j*b.k - a.k*b.j,
65      .j = a.k*b.i - a.i*b.k,
66      .k = a.i*b.j - a.j*b.i
67    };
68    return r;
69  }
70  struct vektor mno i (struct vektor a, long double d) {
71    struct vektor r = {
72      .i = a.i * d,
73      .j = a.j * d,
74      .k = a.k * d
75    };
76    return r;
77  }
78  long double absolutno (struct vektor a) {
79    return sqrtl(a.i*a.i+a.j*a.j+a.k*a.k);
80  }
81  #define MU0 4e-6*M_PI
82  struct vektor tuljava (long double R, unsigned kotov, struct vektor m /* meritev -
        krajevni */) {
83    long double dl_abs = 2*M_PI*R/kotov;  // metri - dol ina vodnika
84    long double dr = 2*M_PI/kotov;    // radiani - kot med dl in to ko na (0,R -
        vrh zanke)
85    struct vektor B = {
86      .i = 0,
87      .j = 0,
88      .k = 0
89    };
90    for (unsigned i = 0; i < kotov; i++) {
91      long double theta = dr*i; // kot na krogu
92      struct vektor dl;
93      dl.j = cosl(theta)*dl_abs;
94      dl.k = -sinl(theta)*dl_abs; // minus po skici sode  ://of.sijanec.eu/sfu/skic
        .jpg
95      dl.i = 0;     // sicer je vseeno, m je na z = 0 in gledamo vse
```

```
96      struct vektor r;
97      r.i = 0;
98      r.j = sinl(theta)*R;
99      r.k = cosl(theta)*R;
100     r = se tej(r, m);
101     B = se tej(B,
102         mno i(
103           vektorski_produkt(dl, r),
104           1/(absolutno(r)*absolutno(r)*absolutno(r))
105         )
106       );
107   }
108   B = mno i(B, MU0/(4*M_PI));
109   return B;
110 } // ena zanka ob toku 1 A. pomno i s tokom in  tevilom  navitij. 0,0 je v
      sredini. B te e v desno.
111 struct vektor zavrti_okoli (struct vektor to ka, struct vektor sredi  e, long
      double kot) {
112   to ka = se tej(to ka, mno i(sredi  e, -1));
113   struct vektor r = {
114     .i = cosl(kot)*to ka.i + sin(kot)*to ka.j,
115     .j = cosl(kot)*to ka.j - sin(kot)*to ka.i
116   };
117   return se tej(r, sredi  e);
118 } // TODO: implement
119 void natisni (FILE * f, struct vektor v, const char * i) {
120   fprintf(f, "vektor %s {\n\t.i = %Lf,\n\t.j = %Lf,\n\t.k = %Lf\n}\n", i, v.i, v.j
      , v.k);
121 }
122 int nari i (long double R, long double I, unsigned n, long double razmak, int
      koliko,
123     unsigned kotov, enum oblika oblika, long double med_tuljavama,
124     struct vektor zamik_izven_osi, long double R2, long double I2, FILE * out) {
125   struct vektor merilno_mesto = { // krajevni vektor
126     .k = 0
127   };
128   if (oblika == PGM)
129     fprintf(out, "P5 %u %u 255\n", koliko*2+1, koliko*2+1);
130   if (oblika == PPM)
131     fprintf(out, "P6 %u %u 255\n", koliko*2+1, koliko*2+1);
132   struct vektor Rpolovic = {
133     .i = med_tuljavama,
134     .j = 0,
135     .k = 0
136   };
137   for (int i = -koliko; i <= koliko; i++) {
138     merilno_mesto.i = i*razmak;
139     for (int j = -koliko; j <= koliko; j++) {
140       merilno_mesto.j = j*razmak;
141       struct vektor merilno_mesto2 = se tej(
142           merilno_mesto,
143           zamik_izven_osi
144         );
145       struct vektor B =
146         se tej(
147           mno i(
148             mno i(
149               tuljava(
150                 R,
151                 kotov,
152                 se tej(
153                   merilno_mesto,
```

```
                  Rpolovic
                )
              ),
              n
            ),
            I
          ),
          mno_i(
            mno_i(
              tuljava(
                R2,
                kotov,
                se_tej(
                  merilno_mesto2,
                  mno_i(
                    Rpolovic,
                    -1
                  )
                )
              ),
              n
            ),
            I2
          )
        );
      switch (oblika) {
        case PGM:
          if (10000*absolutno(B) > 255)
            putc(255, out);
          else
            putc(10000*absolutno(B), out);
          break;
        case PPM:
#define NATISNI_KOMPONENTO(x)   if (10000*fabsl(B.x) > 255) \
            putc(255, out); \
          else \
            putc(10000*fabsl(B.x), out);
          NATISNI_KOMPONENTO(i);
          NATISNI_KOMPONENTO(j);
          NATISNI_KOMPONENTO(k);
          break;
        case TSV:
          fprintf(out, "%Lf\t%Lf\t%Lf\t%Lf\t%Lf\n",
              merilno_mesto.i, merilno_mesto.j,
              B.i, B.j, absolutno(B));
          break;
        case ANIMACIJA:
          abort(); // invalid
          break;
      }
    }
  }
  return 0;
}
int shouldexit = 0;
void handler (int s __attribute__((unused))) {
  shouldexit++;
}
int main (int argc, char ** argv) {
  if (argv[1] && !strcmp(argv[1], "animacija"))
    goto animacija;
  if (argv[1] && !strcmp(argv[1], "enkrat"))
```

```
216     goto enkrat;
217   if (argc < 8)
218     error(1, 0, UVOD, argv[0] ? argv[0] : "./numeri no");
219   long double R = strtold(argv[1], NULL);
220   long double I = strtold(argv[2], NULL);
221   unsigned n = strtol(argv[3], NULL, 10);
222   long double razmak = strtold(argv[4], NULL);
223   int koliko = strtold(argv[5], NULL);
224   unsigned kotov = strtol(argv[6], NULL, 10);
225   enum oblika oblika = PGM;
226   long double med_tuljavama = R/2;
227   struct vektor zamik_izven_osi = {0, 0, 0};
228   long double R2 = R;
229   long double I2 = I;
230   if (argc > 7 && argv[7][0])
231     switch (argv[7][1]) {
232       case 'G':
233       case 'g':
234         oblika = PGM;
235         break;
236       case 'P':
237       case 'p':
238         oblika = PPM;
239         break;
240       case 'S':
241       case 's':
242         oblika = TSV;
243         break;
244       case 'N':
245       case 'n':
246         oblika = ANIMACIJA;
247         break;
248     }
249   if (argc > 8 && argv[8][0])
250     med_tuljavama = strtold(argv[8], NULL);
251   if (argc > 9 && argv[9][0])
252     zamik_izven_osi.j = strtold(argv[9], NULL);
253   if (argc > 0xA && argv[0xA][0])
254     R2 = strtold(argv[0xA], NULL);
255   if (argc > 0xB && argv[0xB][0])
256     I2 = strtold(argv[0xB], NULL);
257   if (oblika != ANIMACIJA)
258     return nari i(R, I, n, razmak, koliko, kotov, oblika, med_tuljavama,
259               zamik_izven_osi, R2, I2, stdout);
260 enkrat:
261   if (argc != 6) {
262     fprintf(stderr, "preveri vnos!\n");
263     return 59;
264   }
265   long double radij = strtold(argv[2], NULL);
266   long double x = strtold(argv[3], NULL);
267   long double y = strtold(argv[4], NULL);
268   unsigned   tevilo_kotov   = strtol(argv[5], NULL, 10);
269   struct vektor Rpolovic = {
270     .i = radij/2,
271     .j = 0,
272     .k = 0
273   };
274   struct vektor merilno_mesto = { // krajevni vektor
275     .i = x,
276     .j = y,
277     .k = 0
```

```
278   };
279   struct vektor B =
280     se tej(
281       mno i(
282         mno i(
283           tuljava(
284             radij,
285               tevilo_kotov  ,
286             se tej(
287               merilno_mesto,
288               Rpolovic
289             )
290           ),
291           1
292         ),
293         1
294       ),
295       mno i(
296         mno i(
297           tuljava(
298             radij,
299               tevilo_kotov  ,
300             se tej(
301               merilno_mesto,
302               mno i(
303                 Rpolovic,
304                 -1
305               )
306             )
307           ),
308           1
309         ),
310         1
311       )
312     );
313   printf("%Lf\t%Lf\t%Lf\t%Lf", B.i, B.j, B.k, absolutno(B));
314   return 0;
315 animacija:
316   signal(SIGTERM, handler);
317   signal(SIGINT, handler);
318   unsigned  as  = 0;
319   while (!shouldexit &&  as  < 21) {
320     char fn[25] = "animacija";
321     sprintf(fn+strlen(fn), "%04d.ppm",  as );
322     fprintf(stderr, "%s\n", fn);
323     FILE * f = fopen(fn, "w");
324     if (!f)
325       error(2, errno, "fopen");
326     long double R = 0.06;
327     long double I = 2;
328     unsigned n = 320;
329     long double razmak = 0.0007;
330     unsigned koliko = 350;
331     unsigned kotov = 30;
332     enum oblika oblika = PPM;
333     long double med_tuljavama = (R/2/10)*( as );
334     struct vektor zamik = {
335       .i = 0,
336       .j = 0, // zamik osi
337       .k = 0
338     };
339     long double R2 = R;
```

```
340     long double I2 = I;
341     if (nari i(R, I, n, razmak, koliko, kotov, oblika, med_tuljavama, zamik, R2,
        I2, f))
342       error(3, errno, "nari i");
343     if (fclose(f))
344       error(4, errno, "fclose");
345      as ++;
346   }
347   return 0;
348 }
```