Vježba 6 - ODREĐIVANJE DOMINANTE RAVNINE NA 2.5D SLICI RANSAC METODOM

Kao cilj ove vježbe postavljeno je primjeniti RANSAC metodu za određivanje dominantne ravnine na 2.5D slikama.

Postupak:

Kako bi mogli pronaći najdominantniju ravninu (i sve ostale ravnine na slici) potrebno je prvo učitati RGB sliku te iz podataka o dubinama izvući svaku dubinu za pojedini piksel. To možemo učiniti pomoću algoritma priloženog u vježbi koji sve podatke sprema u strukturu 3D točke s podacima o poziciji na slici (u pikselima) i pripadajučoj dubini.

Kako bi sada pronašli ravnine na slikama implementiramo RANSAC algoritam:

1. *R ← skup svih piksela slike*
2. *j ← 1*
3. *Ponavljati korake 4 do 14 dokle god je R ≠ ∅*
4. *c ← 0*
5. *Ponavljati korake 6 do 11 odgovarajući broj puta*
6. *Nasumično izabrati 3 piksela m, m' i m'' iz skupa R.*
7. *Odrediti parametre ai, bi i ci ravnine koja prolazi točkama [mT dm] T , [m' T dm'] T i [m'' T dm''] T .*
8. *Odrediti skup W svih piksela skupa R koji zadovoljavaju (9-2).*
9. *Ako je |W| > c onda*
10. *W* ← W*
11. *c ← |W|*
12. *Rj ← W**
13. *j ← j + 1*
14. *R ← R \ Rj*

Za svaki piksel m segmenta Ri vrijedi

$$|[a_i, b_i] \cdot \boldsymbol{m} + c_i - d_m| \leq \varepsilon \qquad (9\text{-}2)$$

gdje su a_i, b_i i c_i parametri ravnine pridružene segmentu R_i, a ε je prag tolerancije koji predstavlja maksimalnu udaljenost točke segmenta od ravnine pridružene tom segmentu.

Postupak sa predavanja profesora Cupeca i Filka (ROBOTSKI VID, 9. 3D Kamere, Robert Cupec, Damir Filko, Osijek, 10.12. 2014.)

Navedeni postupak možemo vidjeti u datoteci *functions.cpp* priloženoj uz ovaj dokument. Ravnine se pronalaze redom od najdominantnije do najmanje dominantne ravnine. Kako je algoritam implementiran na takav način, kada želimo izabrati najdominantniju ravninu možemo odabrati samo prvi element iz vektora svih ravnina.

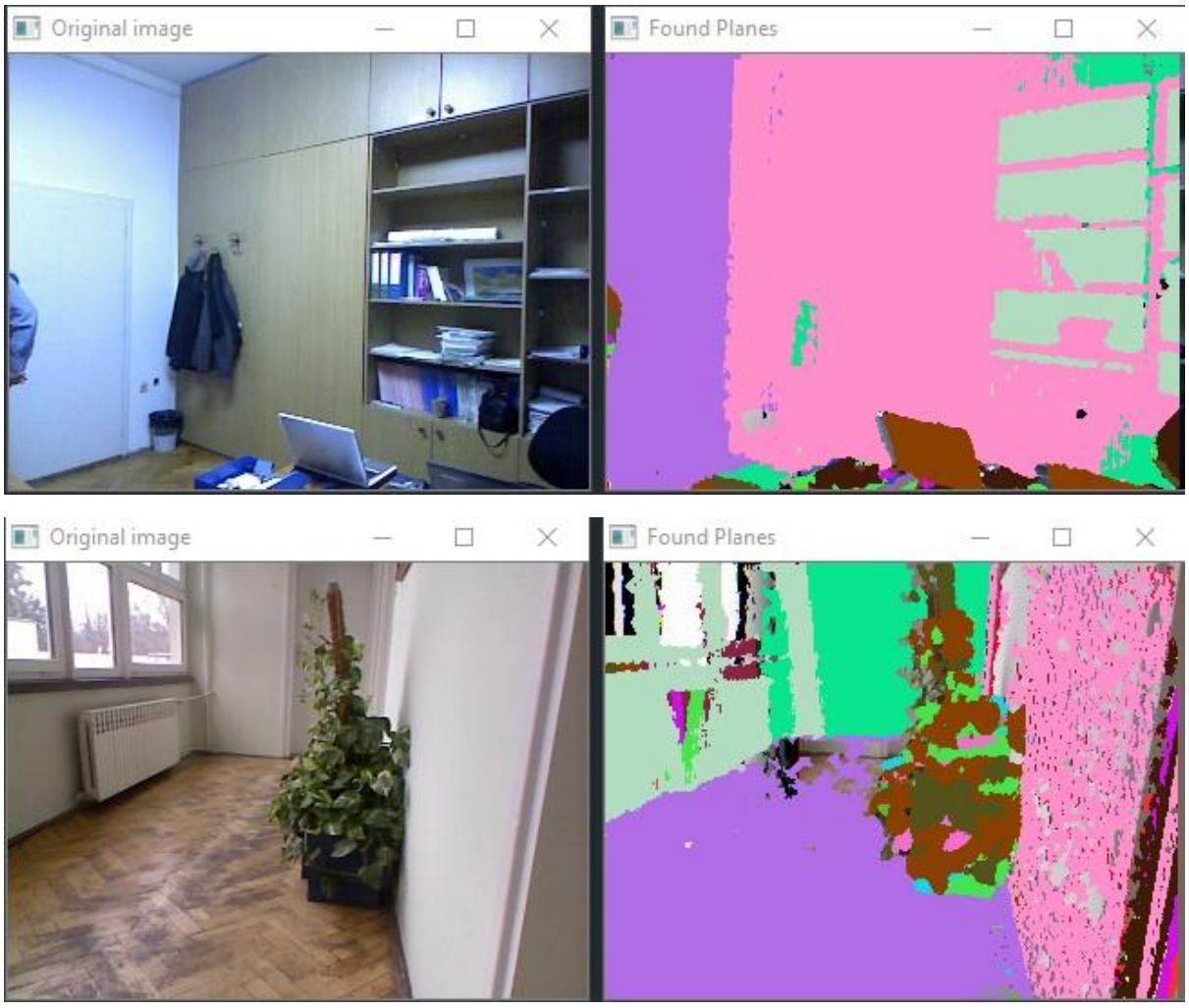
Kako je RANSAC algoritam iterativan, potrebna je znatna količina vremena da se izvede ukoliko tražimo sve ravnine na slici. Iako se pokazalo da odabrani broj iteracije ne utječe bitno

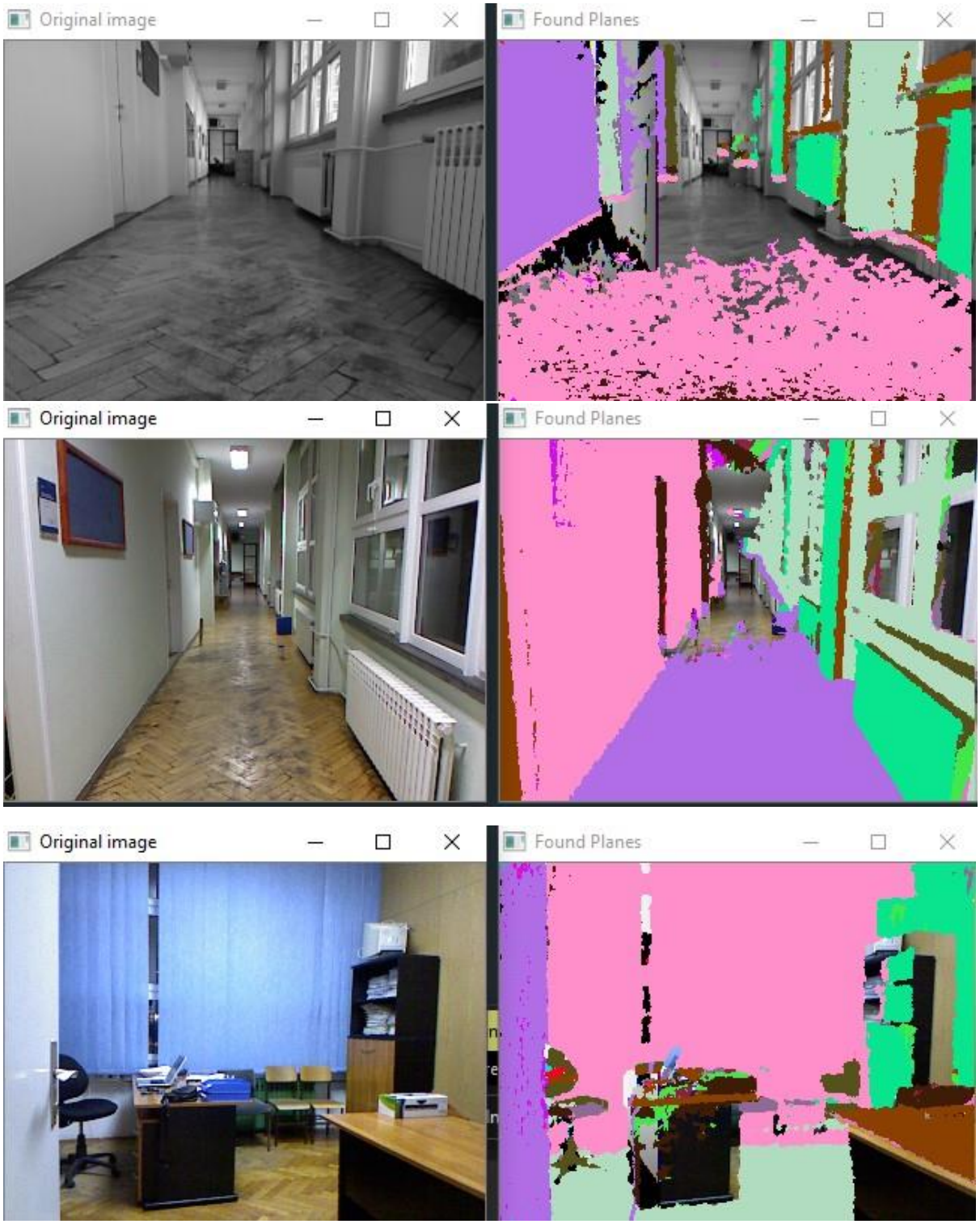
na brzinu izvođenja (naravno, unutar nekih razumnih okvira promjena broja iteracija), kao glavni problem nametnulo se izbacivanje točaka pronađene najdominantnije ravnine iz skupa svih točaka. Implementacijom nekoliko algoritama (vlastitih i postojećih u c++) možemo vidjeti da svi imaju relativno jednaku brzinu izvođenja, odnosno oslanjaju se na isti iterativni postupak pretraživanja vektora svih točaka slike, što je jako vremenski zahtjevno. Odabrani način prikazan u funkciji *getAllPlanes* izabran je zbog svoje lake čitljivosti i jednostavnosti za razumjeti, uz overload operatora == kako bi mogao raditi sa strukturom točaka RGB-D slike.

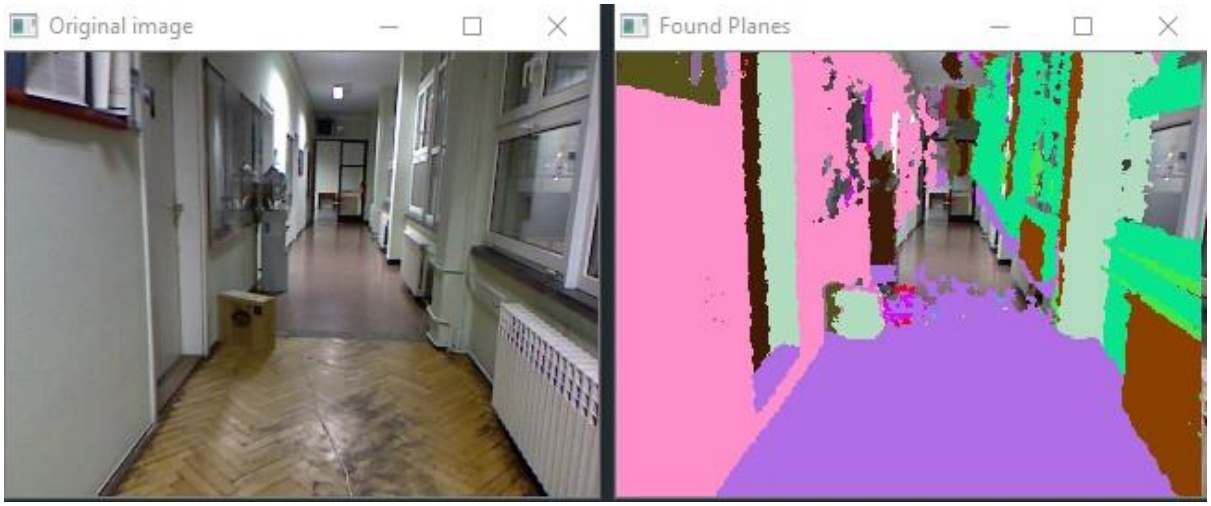
Ukoliko je nužno osigurati brže izvođenje programa pri pronalasku samo jedne ravnine, možemo nakon prvog pronalaska (najdominantnije) ravnine odmah izaći iz funkcije te vratiti samo parametre pronađene ravnine. Algoritam priložen u navedenoj funkciji pronalazi sve ravnine pa naknadno ispisuje samo najdominantniju.

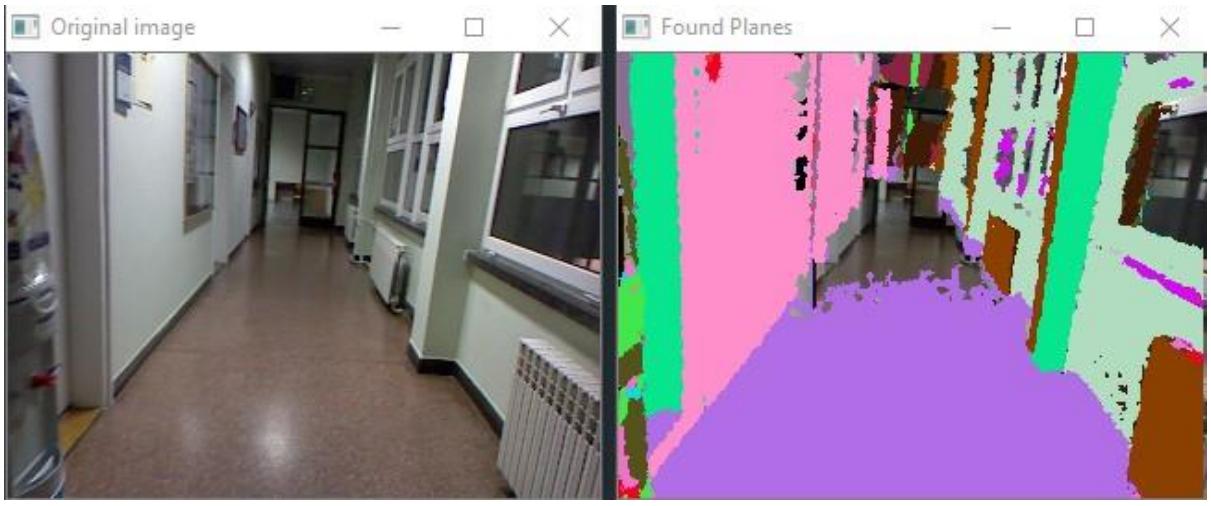
Rezultati u obje metode će biti isti jer se ravnine prepoznaju redom od one kojoj pripada najviše točaka slike do one kojoj pripada najmanje.

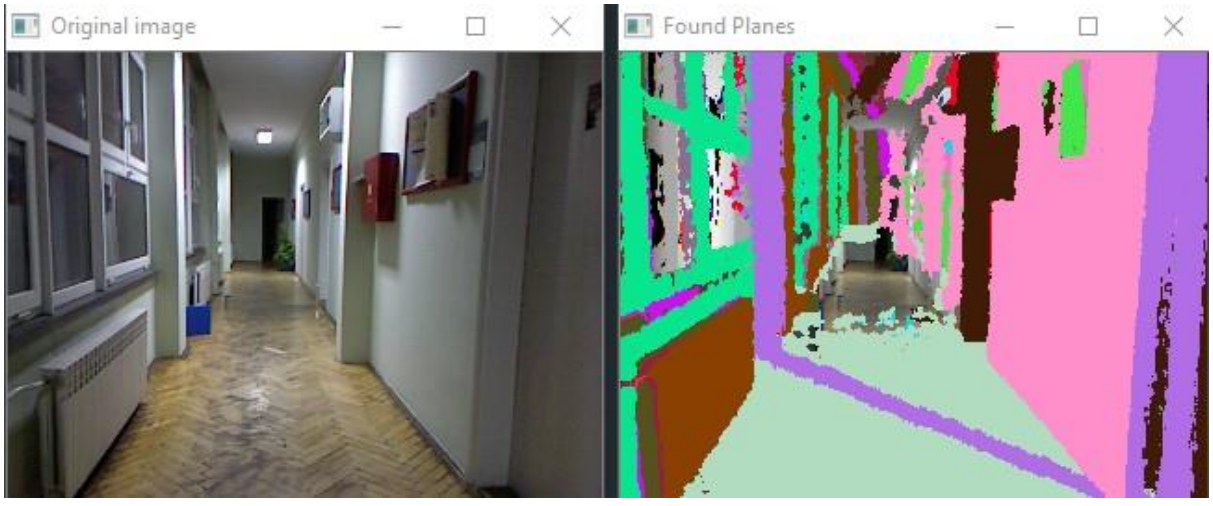
Rezultati:

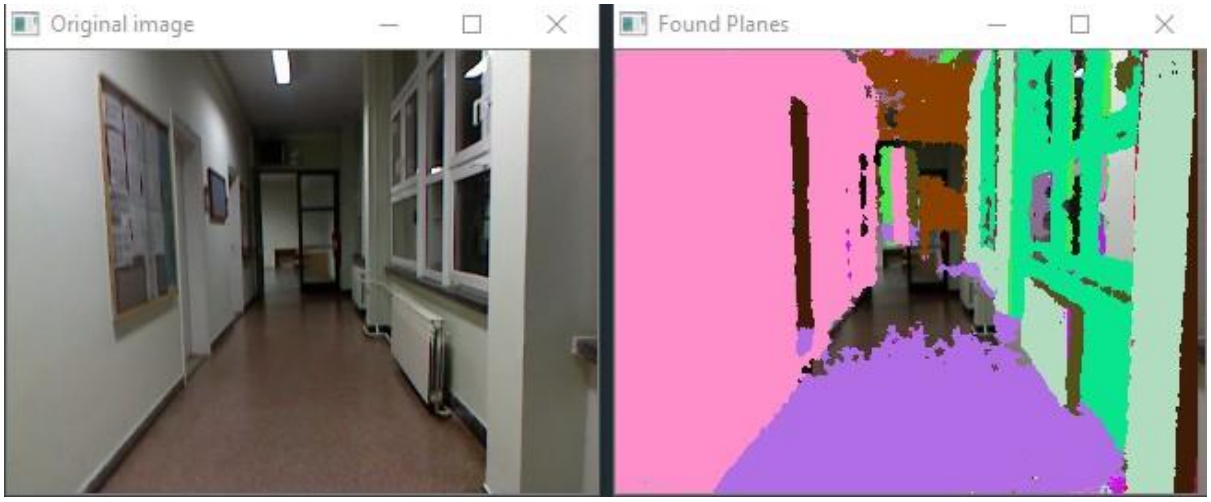



Original image
Found Planes
Original image
Found Planes
Original image
Found Planes

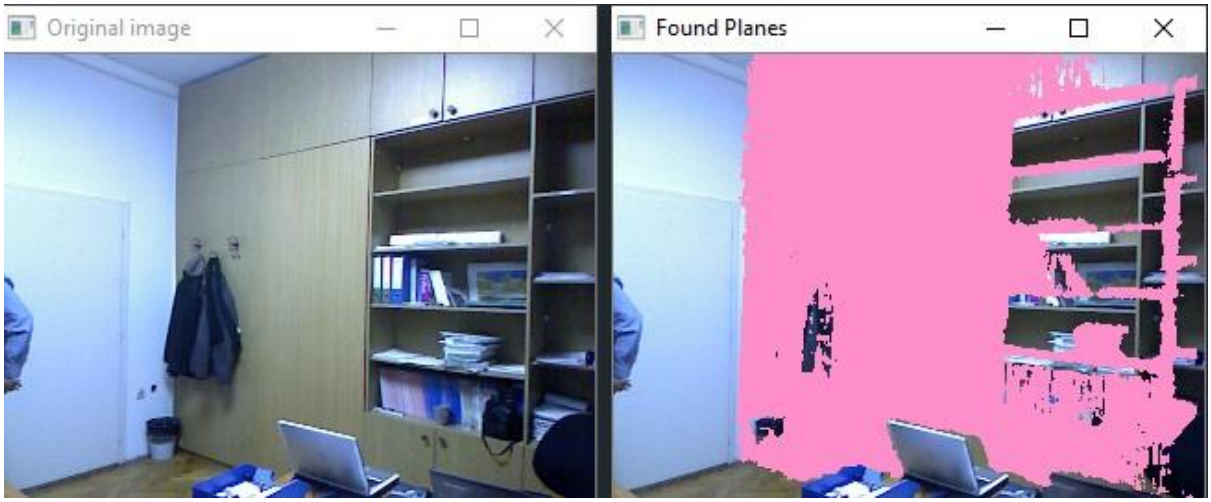

Original image
Found Planes

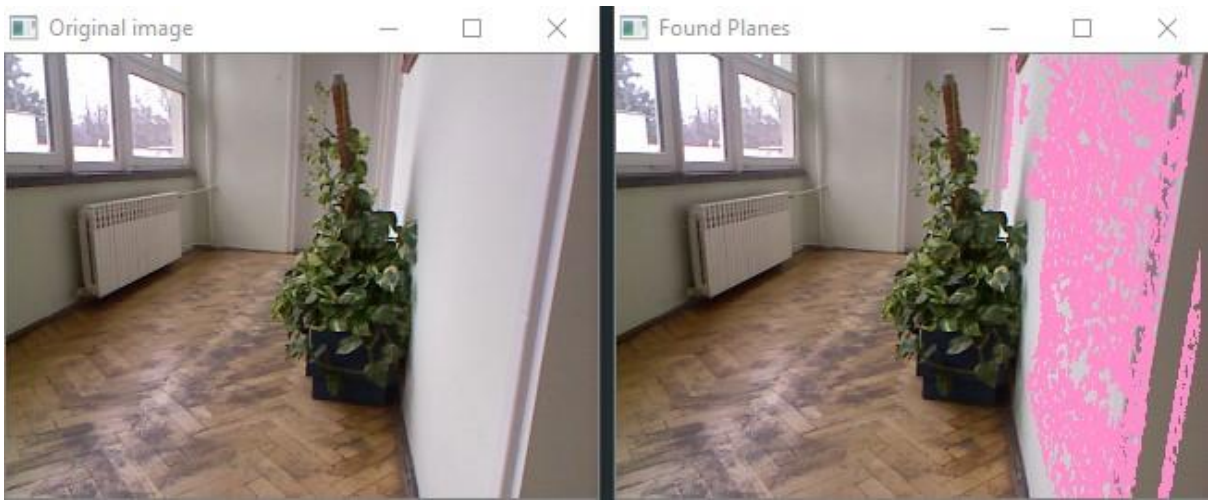

Original image
Found Planes

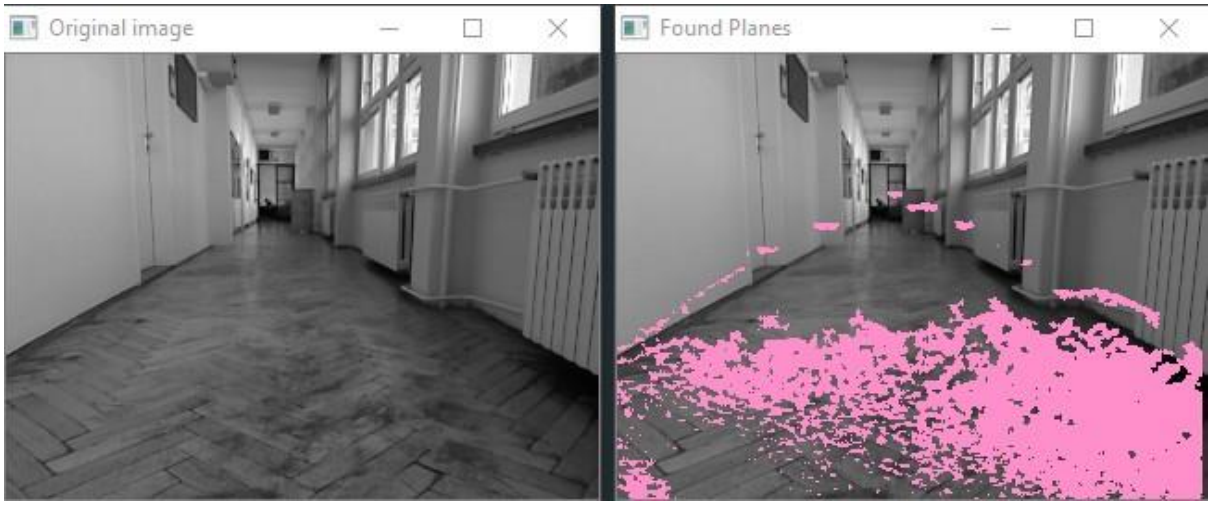

Original image
Found Planes

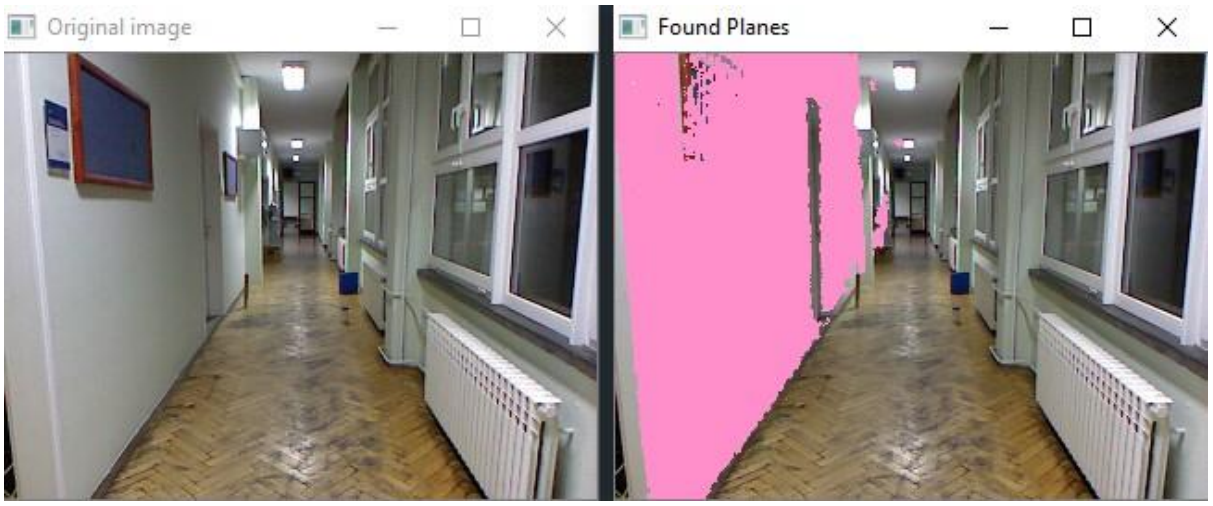


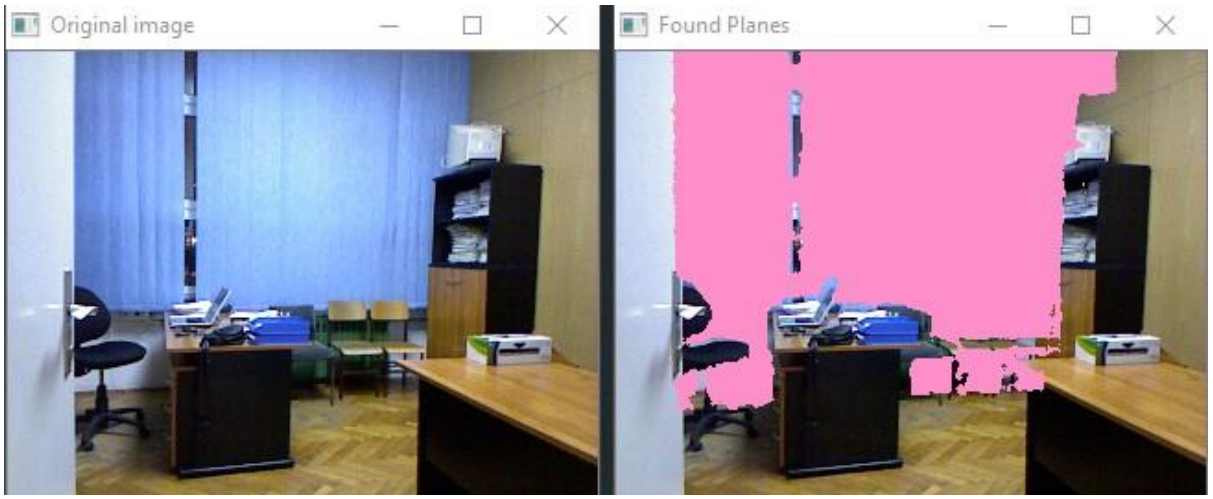
Detektirana najdominantnija ravnina:



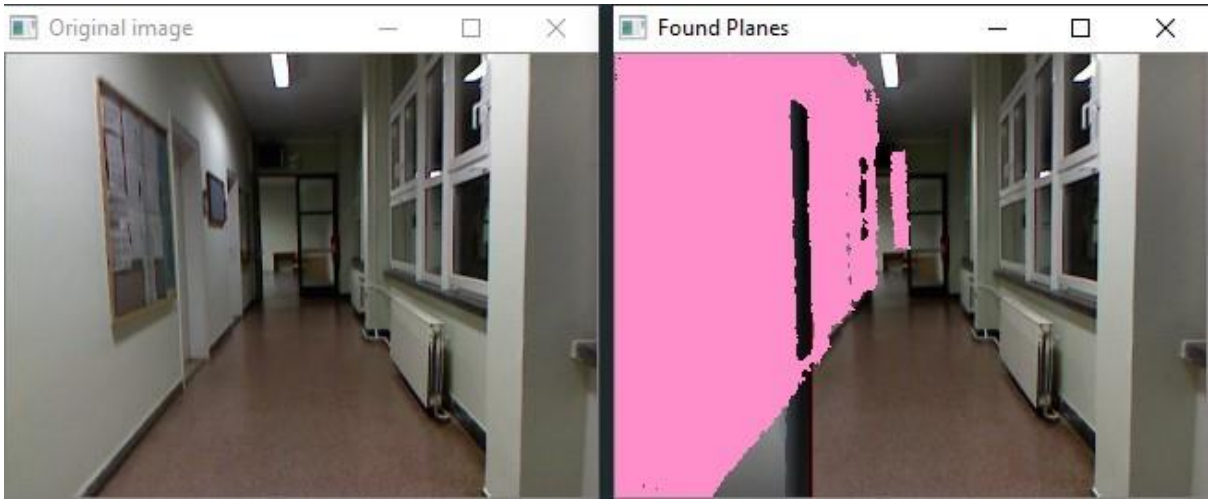



Original image
Found Planes


Original image
Found Planes


Original image
Found Planes


Original image
Found Planes
Original image
Found Planes
Original image
Found Planes


Original image
Found Planes