```
using Images, LinearAlgebra, Plots
```

Enable webcam

```
@bind raw_camera_data camera_input()
```

p =

```
p = load(joinpath(@__DIR__, "Estelada.png"))
#p = load(joinpath(@__DIR__, "MapleLeaf.png"))
#p = load(joinpath(@__DIR__, "StarsAndStripes.gif"))
#p = load(joinpath(@__DIR__, "Communism.jpg"))
#p = process_raw_camera_data(raw_camera_data)
```

```
Array{RGB{Normed{UInt8,8}},2}
```

```
typeof(p)
```

```
(170, 255)
```

```
size(p)
```

```
begin
    m,n = size(p);
    R = Float64[p[i,j].r for i in 1:m, j in 1:n]
    G = Float64[p[i,j].g for i in 1:m, j in 1:n]
    B = Float64[p[i,j].b for i in 1:m, j in 1:n]
end;
```

```
170×255 Array{Float64,2}:
 0.5607843137254902    …  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705  …  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 ⋮                     ⋱                 ⋮
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705  …  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.058823529411764705     1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 0.5294117647058824       1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
```

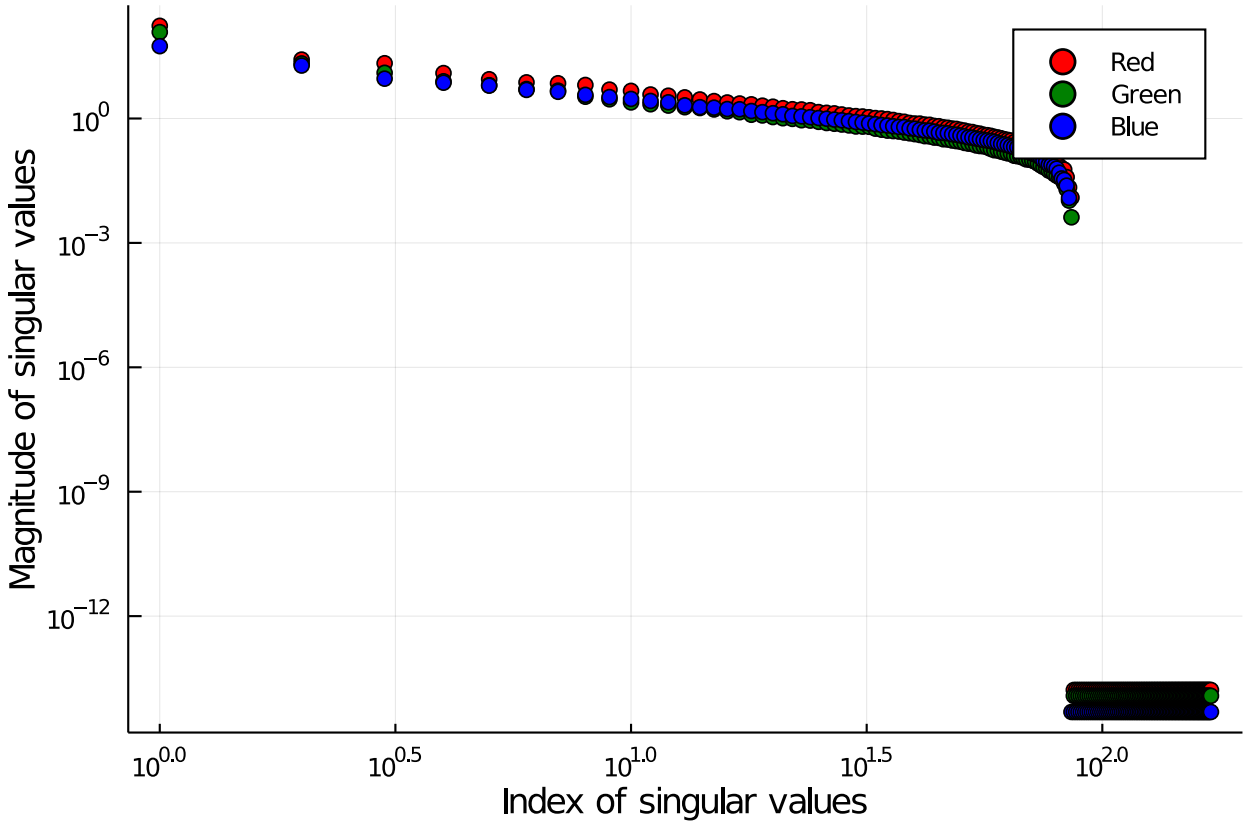
```
R
```

```
SVD{Float64,Float64,Array{Float64,2}}
U factor:
170×170 Array{Float64,2}:
 0.0008214383047462637  0.0029014555531801673  …   0.0
 0.002634348608926307   0.00929784343935569        0.03768164089348047
 0.004386354327904685   0.015457636301031696      -0.028059931277411097
 0.0061370217170655     0.021576817546556937      -0.12583709064134968
 0.007885816626590257   0.02763931034276909       -0.02705863848840455
 0.00963220547797677    0.03362918679321816   …    0.028053685460237676
 0.011375655426840336   0.039530709785291475       0.11934724599503621
 ⋮                                            ⋱
 0.00963220547797677    0.033629186793218156      -0.0588549114178476
 0.007885816626590257   0.027639310342769063  …    0.05750985051467486
 0.006137021717065493   0.021576817546557048       0.09573191511537926
 0.004386354327904699   0.015457636301031578       0.05782300253604286
 0.0026343486089262643  0.009297843439355255      -0.06710649543326379
 0.0008815435465569655  0.0031137571790226855      0.02909048119273905
singular values:
170-element Array{Float64,1}:
 55.56055571087819
 18.928517587911564
  9.134814930734203
  7.306141113506519
  6.208429719051431
  4.89548197279963
  4.409809902673164
  ⋮
  4.885956032098706e-15
  4.885956032098706e-15
  4.885956032098706e-15
  4.885956032098706e-15
  4.885956032098706e-15
  4.885956032098706e-15
Vt factor:
170×255 Array{Float64,2}:
  0.1419279270359209   …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.1707885899947962      0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.18484355045854242     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.06355875550891697     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
 -0.16258457258231562     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.07075537792967342  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
 -0.20497303358503485     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  ⋮                    ⋱                ⋮
  0.0                     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.0                  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.0                     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.0                     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.0                     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
  0.0                     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

```
begin
    UR,ΣR,VR = svd(R);
    UG,ΣG,VG = svd(G);
    UB,ΣB,VB = svd(B);
end
```

(61, 44, 53)

```
begin
    σ = ΣR[1] + ΣG[1] + ΣB[1]
    findfirst(i -> i < 1e-3*σ, ΣR),findfirst(i -> i < 1e-3*σ, ΣG),findfirst(i -> i < 1e-3*σ, ΣB)
end
```




```
begin
    scatter(1:length(ΣR), ΣR; xscale=:log10, yscale=:log10, color=:red, label="Red")
    scatter!(1:length(ΣG), ΣG; color=:green, label="Green")
    scatter!(1:length(ΣB), ΣB; color=:blue, label="Blue")
    xlabel!("Index of singular values")
    ylabel!("Magnitude of singular values")
end
```

r =

```
begin
    RR = UR[:,1:r]*(ΣR[1:r].*VR[:,1:r]')
    GG = UG[:,1:r]*(ΣG[1:r].*VG[:,1:r]')
    BB = UB[:,1:r]*(ΣB[1:r].*VB[:,1:r]')
    p[:] = [RGB{Normed{UInt8,8}}(clamp(RR[i,j], 0.f0, 1.f0), clamp(GG[i,j], 0.f0, 1.f0),
clamp(BB[i,j], 0.f0, 1.f0)) for i in 1:m, j in 1:n]
end
```

rR = rG = rB =

```
begin
    RRR = UR[:,1:rR]*(ΣR[1:rR].*VR[:,1:rR]')
    GGG = UG[:,1:rG]*(ΣG[1:rG].*VG[:,1:rG]')
    BBB = UB[:,1:rB]*(ΣB[1:rB].*VB[:,1:rB]')
    p[:] = [RGB{Normed{UInt8,8}}(clamp(RRR[i,j], 0.f0, 1.f0), clamp(GGG[i,j], 0.f0, 1.f0),
clamp(BBB[i,j], 0.f0, 1.f0)) for i in 1:m, j in 1:n]
end
```

```
camera_input (generic function with 1 method)
```

```
process_raw_camera_data (generic function with 1 method)
```