

JANICA KOTELIC - TRAININGSLÄUFE



Janica Kostelić ist eine der erfolgreichsten Skirennläuferinnen aller Zeiten und eine Legende des alpinen Skisports. Die kroatische Athletin wurde am 5. Januar 1982 in Zagreb geboren und erlangte weltweite Bekanntheit durch ihre beeindruckenden Leistungen auf der Skipiste.

Janica gewann insgesamt vier olympische Goldmedaillen und eine Silbermedaille, alle bei den Olympischen Winterspielen 2002 in Salt Lake City und 2006 in Turin. Damit ist sie die erfolgreichste Skirennläuferin in der Geschichte der Olympischen Spiele.

Neben ihren olympischen Erfolgen sicherte sie sich auch fünf Weltmeistertitel und gewann dreimal den Gesamtweltcup (2001, 2003, 2006). Trotz zahlreicher Verletzungen zeigte sie außergewöhnliche Entschlossenheit und bewies, dass sie zu den besten Skirennläuferinnen der Geschichte gehört. Ihre Erfolge haben den Skisport in Kroatien populär gemacht und sie zur nationalen Heldin gemacht.



Die Idee wahr verschiedenen Trainingsläufe zu scripten die randomisiert erstellt werden für die Kroatische Skirennläuferin.

Am Anfang wurden verschieden curven gscriptet um eine ähnliche art von Skiläufern zu erstellen. Weiter habe ich Stock-Markierungen gscriptet und witer wird das auf den Dachstein Surface projiziert die weiter bearbeitet wurden um die Ähnlichkeit zu einem echten Trainingslauf ahnen.

```
23 dm.newEmptyLayer("Default")
24 dm.newEmptyLayer("Projected_Curves")
25 dm.newEmptyLayer("Projected_Curves::curves", [210,210,190])
26 dm.newEmptyLayer("Projected_Curves::szfs", [10,10,40])
27 rs.LayerMaterialIndex("Projected_Curves::szfs", rs.LayerMaterialIndex("GIS:DOM_5m"))
28 dm.newEmptyLayer("Adjusted_Curves", [0,200,0])
29 # Remove existing curves from the "Adjusted_Curves" layer
30
31 def delete_adjusted_curves():
32     objects = rs.ObjectByLayer("Adjusted_Curves")
33     if objects:
34         rs.DeleteObjects(objects)
35
36 #delete_adjusted_curves()
37
38 dm.newEmptyLayer("PO3", [0, 0, 200])
39
40 dist = 40
41 anz = 11
42 anzTweens = 16
43 anzRuns = 8
44
45 for i in range(anzRuns):
46     start = [1100 + dist * i, 1600, 0]
47     ziel = rs.VectorAdd(start, [random.uniform(40, 120), random.uniform(-300, -300) - i * 10, 0])
48     lin = rs.AddCurve([start, ziel])
49     coords = rs.DivideCurve(lin, anz, 0)
50     rs.DeleteObject(lin)
51
52     if i:
53         coordsX = []
54         for j in range(0, anz - 1):
55             coordsX.append(rs.VectorAdd(coords[j], [random.uniform(-40, 40), random.uniform(-40, 20), 0]))
56         coordsX.append(ziel)
57         for deg in [2,7]:
58             crv = rs.AddCurve(coordsX, deg)
59             rs.DeleteObject(crv)
60             crv = rs.CopyObject(crv)
61             rs.RebuildCurve(crv, degree=deg, point_count=anz)
62             rs.ObjectName(crv, "slalom_"+str(i)+"_"+str(deg))
63             tweens = rs.AddTweenCurves(rs.AllObjects()[0], rs.AllObjects()[1], number_of_curves=anzTweens, method=1, sample_number=10)
64             rs.ObjectName(tweens, "slalom_"+str(i)+"_0")
65
66 if i:
67     dm.newEmptyLayer("Stoecke", [200, 0, 0])
68     poleSpacing = 2
69     lineThickness = 5
70
71     for crv in rs.ObjectsByName("slalom_"+str(deg)):
72         crv_name = "slalom_"+str(deg)
73         rs.DeleteObject(crv)
74         crv = rs.CopyObject(crv)
75         coords = rs.CurveEditPoints(crv)
76
77         for j in range(1, len(coords), 2):
78             base = coords[j]
79             offsetV = [poleSpacing, 0, 0]
80             secondPoint = rs.VectorAdd(base, offsetV)
81
82             line1 = rs.AddLine(base, rs.VectorAdd(base, [0, 0, poleSpacing*2]))
83             line2 = rs.AddLine(secondPoint, rs.VectorAdd(secondPoint, [0, 0, poleSpacing*2]))
84
85             rs.EnablePrintPreview to see
86             rs.ObjectPrintWidth(line1, lineThickness)
87             rs.ObjectPrintWidth(line2, lineThickness)
88
89             rs.ObjectColor(line1, (255, 0, 0))
90             rs.ObjectColor(line2, (0, 255, 0))
91
92     #####
93     ### PROJECTING
94     model_surface = rs.ObjectsByName("040_3")[0]
95     anzR = 64*4
96     rs.CurrentLayer("Projected_Curves::szfs")
97
98     #####GoGo
99     for run in range(0, anzRuns, 1):
100         slaloms = rs.ObjectsByName("slalom_"+str(run)+"_")
101         for crv in slaloms:
102             nam = rs.ObjectName(crv)
103             projected = rs.ProjectCurveToSurface(crv, model_surface, [0,0,1])[0]
104             rs.RebuildCurve(projected, 1, anzR)
105             rs.ObjectName(projected, nam)
106             rs.DeleteObject(crv)
107             rs.Redraw()
108
109     #####
110     rs.CurrentLayer("Stoecke")
111     all_top_points = []
112     pole_centers = []
113     stockKlev_list = []
114
115     stoeckeUnten = rs.ObjectsByLayer("Stoecke")
116     for stock in stoeckeUnten:
117         pt0 = rs.CurveStartPoint(stock)
118         ptX = rs.AddPoint(pt0)
119         ptF = rs.ProjectPointToSurface(ptX, model_surface, [0,0,1])[0]
120         rs.DeleteObject(ptX)
121
122         ptOben = rs.VectorAdd(ptF, [0,0,7])
123         ptOben = rs.VectorAdd(ptOben, dm.randVec(-1,1))
124
125         stockKlev = rs.AddLine(ptF, ptOben)
126         rs.ObjectColor(stockKlev, [0,0,175])
127         rs.ObjectLayer(stockKlev, "Stoecke")
128
129         IF = rs.DivideCurve(stockKlev, 3, create_points=False)
130         if IF and len(IF) > 1:
131             mid_index = len(IF) // 2
132             all_top_points.append(IF[mid_index])
133             pole_centers.append(ptF)
134             stockKlev_list.append(stockKlev)
135
136     rs.DeleteObjects(stoeckeUnten)
137
138     paired_indices = set()
139     pairs = []
140
141     for i, base_pt in enumerate(pole_centers):
142         if i in paired_indices:
143             continue
144         min_dist = float("inf")
145         closest = None
146         for j, other_pt in enumerate(pole_centers):
147             if i != j and j not in paired_indices:
148                 dist = rs.Distance(base_pt, other_pt)
149                 if dist < min_dist:
150                     min_dist = dist
151                     closest = j
```

CODE / DACHSTEIN 3D / RANDOM / CURVES /
LINES / REBUILD