File - performance_get_output_generator

```
C:\Users\usuario\Anaconda3\envs\tfq\python.exe C:/Users/usuario/qGAN/quantumGAN/performance_testing/
performance_get_output_generator.py
[0.08984375 0.76904297 0.11279297 0.02832031]
[0.10888672 0.78027344 0.09179688 0.01904297]
[0.08642578 0.77587891 0.11376953 0.02392578]
[0.09375    0.77832031 0.10986328 0.01806641]
[0.10009766 0.76171875 0.10742188 0.03076172]
[0.08837891 0.77050781 0.11328125 0.02783203]
[0.09716797 0.76611328 0.11132812 0.02539062]
[0.10351562 0.77441406 0.09716797 0.02490234]
[0.09521484 0.78369141 0.09667969 0.02441406]
[0.10888672 0.75097656 0.11230469 0.02783203]
[0.09033203 0.78076172 0.10449219 0.02441406]
[0.10253906 0.76611328 0.11181641 0.01953125]
[0.10498047 0.77197266 0.10302734 0.02001953]
[0.09228516 0.76513672 0.11376953 0.02880859]
[0.10546875 0.76220703 0.10791016 0.02441406]
[0.09521484 0.78613281 0.09570312 0.02294922]
[0.08447266 0.77783203 0.10986328 0.02783203]
[0.09765625 0.77294922 0.10498047 0.02441406]
[0.09814453 0.76660156 0.11523438 0.02001953]
[0.08789062 0.78271484 0.10888672 0.02050781]
[0.09716797 0.78710938 0.09765625 0.01806641]
[0.09570312 0.77294922 0.10595703 0.02539062]
[0.09472656 0.76660156 0.11376953 0.02490234]
[0.08886719 0.78613281 0.09814453 0.02685547]
[0.08837891 0.77880859 0.10839844 0.02441406]
[0.10351562 0.76171875 0.10839844 0.02636719]
[0.0859375  0.77539062 0.11572266 0.02294922]
[0.08886719 0.77929688 0.11181641 0.02001953]
[0.08300781 0.78466797 0.10449219 0.02783203]
[0.09326172 0.78271484 0.10302734 0.02099609]
[0.09472656 0.77636719 0.10595703 0.02294922]
[0.09912109 0.76318359 0.10546875 0.03222656]
[0.09912109 0.76416016 0.10888672 0.02783203]
[0.09423828 0.77539062 0.10205078 0.02832031]
[0.08398438 0.77539062 0.11621094 0.02441406]
[0.09960938 0.74658203 0.11962891 0.03417969]
[0.09423828 0.77392578 0.11035156 0.02148438]
[0.08886719 0.79980469 0.08935547 0.02197266]
[0.10302734 0.77099609 0.09667969 0.02929688]
[0.08691406 0.78857422 0.09716797 0.02734375]
[0.08105469 0.77978516 0.11816406 0.02099609]
[0.10058594 0.75195312 0.12060547 0.02685547]
[0.09765625 0.77294922 0.10205078 0.02734375]
[0.08349609 0.78857422 0.10205078 0.02587891]
[0.08935547 0.7890625  0.10302734 0.01855469]
[0.09375    0.77197266 0.10498047 0.02929688]
[0.09423828 0.77783203 0.10791016 0.02001953]
[0.11230469 0.76269531 0.09814453 0.02685547]
[0.09277344 0.77148438 0.11279297 0.02294922]
[0.10546875 0.74462891 0.12011719 0.02978516]
[0.09228516 0.76953125 0.10449219 0.03369141]
[0.09082031 0.77148438 0.11083984 0.02685547]
[0.09179688 0.77880859 0.10595703 0.0234375 ]
[0.09375    0.765625   0.11376953 0.02685547]
[0.09326172 0.78271484 0.1015625  0.02246094]
[0.08496094 0.79394531 0.1015625  0.01953125]
[0.09472656 0.76171875 0.11816406 0.02539062]
[0.09228516 0.77441406 0.109375   0.02392578]
[0.09863281 0.76806641 0.10888672 0.02441406]
[0.10205078 0.77587891 0.10058594 0.02148438]
[0.09667969 0.76904297 0.10693359 0.02734375]
[0.10107422 0.76611328 0.10742188 0.02539062]
[0.08398438 0.765625   0.12255859 0.02783203]
[0.09960938 0.77050781 0.10644531 0.0234375 ]
[0.10009766 0.76806641 0.10693359 0.02490234]
[0.10302734 0.765625   0.09863281 0.03271484]
[0.08886719 0.77539062 0.11083984 0.02490234]
[0.09130859 0.77001953 0.11572266 0.02294922]
[0.10107422 0.76953125 0.10791016 0.02148438]
[0.09667969 0.77783203 0.10400391 0.02148438]
[0.09814453 0.77099609 0.11279297 0.01806641]
[0.09375    0.78613281 0.09960938 0.02050781]
[0.08496094 0.78369141 0.10693359 0.02441406]
[0.09521484 0.76171875 0.10839844 0.03466797]
[0.10595703 0.75732422 0.11230469 0.02441406]
[0.10449219 0.76220703 0.11083984 0.02246094]
[0.08496094 0.77880859 0.11425781 0.02197266]
[0.07958984 0.77832031 0.12109375 0.02099609]
[0.09716797 0.76806641 0.11035156 0.02441406]
[0.09423828 0.77099609 0.11279297 0.02197266]
[0.09375    0.77050781 0.10791016 0.02783203]
[0.08886719 0.77490234 0.10839844 0.02783203]
[0.08740234 0.77832031 0.10986328 0.02441406]
[0.09423828 0.77050781 0.10742188 0.02783203]
[0.09130859 0.77441406 0.11328125 0.02099609]
[0.10644531 0.75927734 0.10498047 0.02929688]
[0.10058594 0.76660156 0.10498047 0.02783203]
```

```
 89 [0.09326172 0.77294922 0.11230469 0.02148438]
 90 [0.08496094 0.79541016 0.09570312 0.02392578]
 91 [0.09375    0.78466797 0.09472656 0.02685547]
 92 [0.09179688 0.78662109 0.09423828 0.02734375]
 93 [0.09228516 0.77050781 0.10888672 0.02832031]
 94 [0.10449219 0.76269531 0.10742188 0.02539062]
 95 [0.09667969 0.765625   0.10791016 0.02978516]
 96 [0.09228516 0.78466797 0.09619141 0.02685547]
 97 [0.09033203 0.77929688 0.10449219 0.02587891]
 98 [0.11132812 0.74560547 0.11767578 0.02539062]
 99 [0.08251953 0.77441406 0.109375   0.03369141]
100 [0.09375    0.76855469 0.11425781 0.0234375 ]
101 [0.09863281 0.77197266 0.10449219 0.02490234]
102 [0.09130859 0.77392578 0.10693359 0.02783203]
103 [0.08691406 0.77587891 0.109375   0.02783203]
104 [0.09033203 0.78466797 0.10107422 0.02392578]
105 [0.09765625 0.76611328 0.10839844 0.02783203]
106 [0.09619141 0.75927734 0.12011719 0.02441406]
107 [0.09619141 0.76660156 0.11083984 0.02636719]
108 [0.09277344 0.76416016 0.11767578 0.02539062]
109 [0.0859375  0.78955078 0.10253906 0.02197266]
110 [0.08984375 0.77294922 0.11376953 0.0234375 ]
111 [0.08642578 0.77685547 0.11230469 0.02441406]
112 [0.09277344 0.76806641 0.109375   0.02978516]
113 [0.10595703 0.75976562 0.10839844 0.02587891]
114 [0.10400391 0.76611328 0.10009766 0.02978516]
115 [0.09960938 0.77246094 0.10644531 0.02148438]
116 [0.09082031 0.78076172 0.10791016 0.02050781]
117 [0.09277344 0.78613281 0.09765625 0.0234375 ]
118 [0.09570312 0.77734375 0.10302734 0.02392578]
119 [0.09082031 0.77783203 0.10742188 0.02392578]
120 [0.08935547 0.78320312 0.10107422 0.02636719]
121 [0.09521484 0.77148438 0.10498047 0.02832031]
122 [0.09082031 0.76855469 0.11572266 0.02490234]
123 [0.10498047 0.75097656 0.11816406 0.02587891]
124 [0.08544922 0.78125    0.11132812 0.02197266]
125 [0.09667969 0.76708984 0.11523438 0.02099609]
126 [0.09521484 0.76611328 0.10791016 0.03076172]
127 [0.09863281 0.77734375 0.10400391 0.02001953]
128 [0.09326172 0.77246094 0.10595703 0.02832031]
129 [0.09619141 0.78076172 0.09765625 0.02539062]
130 [0.10888672 0.76318359 0.10498047 0.02294922]
131 [0.08740234 0.77587891 0.11621094 0.02050781]
132 [0.08740234 0.76953125 0.11816406 0.02490234]
133 [0.10449219 0.76416016 0.10644531 0.02490234]
134 [0.10009766 0.77978516 0.09619141 0.02392578]
135 [0.10253906 0.77929688 0.09814453 0.02001953]
136 [0.08935547 0.78710938 0.09765625 0.02587891]
137 [0.09082031 0.77539062 0.11230469 0.02148438]
138 [0.10058594 0.765625   0.10986328 0.02392578]
139 [0.09619141 0.76806641 0.10839844 0.02734375]
140 [0.10009766 0.75878906 0.11865234 0.02246094]
141 [0.09667969 0.76953125 0.10742188 0.02636719]
142 [0.09228516 0.765625   0.11425781 0.02783203]
143 [0.08642578 0.78466797 0.10693359 0.02197266]
144 [0.09130859 0.78613281 0.09912109 0.0234375 ]
145 [0.09570312 0.76513672 0.11816406 0.02099609]
146 [0.08691406 0.77197266 0.11132812 0.02978516]
147 [0.09814453 0.77539062 0.10107422 0.02539062]
148 [0.09960938 0.76367188 0.11083984 0.02587891]
149 [0.10107422 0.76660156 0.11523438 0.01708984]
150 [0.09033203 0.76855469 0.11669922 0.02441406]
151 [0.10107422 0.77148438 0.10253906 0.02490234]
152 [0.08886719 0.76904297 0.11523438 0.02685547]
153 [0.09814453 0.77246094 0.10058594 0.02880859]
154 [0.08935547 0.77734375 0.11132812 0.02197266]
155 [0.09326172 0.76757812 0.11230469 0.02685547]
156 [0.09326172 0.78857422 0.09960938 0.01855469]
157 [0.09619141 0.78173828 0.10205078 0.02001953]
158 [0.08691406 0.77880859 0.10839844 0.02587891]
159 [0.09667969 0.76171875 0.11865234 0.02294922]
160 [0.09716797 0.77246094 0.11083984 0.01953125]
161 [0.09619141 0.76904297 0.09960938 0.03515625]
162 [0.09082031 0.77685547 0.11035156 0.02197266]
163 [0.09912109 0.77197266 0.09960938 0.02929688]
164 [0.10205078 0.75439453 0.11767578 0.02587891]
165 [0.07763672 0.78613281 0.09960938 0.03662109]
166 [0.10742188 0.76025391 0.10986328 0.02246094]
167 [0.09326172 0.78369141 0.09960938 0.0234375 ]
168 [0.08740234 0.78662109 0.10839844 0.01757812]
169 [0.10498047 0.77099609 0.10839844 0.015625  ]
170 [0.10400391 0.76416016 0.109375   0.02246094]
171 [0.09814453 0.75830078 0.11816406 0.02539062]
172 [0.09863281 0.79296875 0.08496094 0.0234375 ]
173 [0.09130859 0.77587891 0.10449219 0.02832031]
174 [0.10058594 0.76757812 0.11083984 0.02099609]
175 [0.10791016 0.77050781 0.09765625 0.02392578]
176 [0.09228516 0.77636719 0.10498047 0.02636719]
177 [0.09570312 0.78271484 0.09912109 0.02246094]
```

```
178 [0.09033203 0.78320312 0.10351562 0.02294922]
179 [0.09082031 0.78076172 0.10058594 0.02783203]
180 [0.09277344 0.77441406 0.11083984 0.02197266]
181 [0.09179688 0.77880859 0.10107422 0.02832031]
182 [0.08886719 0.77587891 0.11230469 0.02294922]
183 [0.10742188 0.76416016 0.09765625 0.03076172]
184 [0.09130859 0.77587891 0.10791016 0.02490234]
185 [0.09814453 0.77148438 0.10449219 0.02587891]
186 [0.09863281 0.76904297 0.10791016 0.02441406]
187 [0.09863281 0.76464844 0.11035156 0.02636719]
188 [0.10742188 0.76074219 0.11083984 0.02099609]
189 [0.08544922 0.77392578 0.11132812 0.02929688]
190 [0.08691406 0.77685547 0.11083984 0.02539062]
191 [0.09863281 0.78857422 0.08984375 0.02294922]
192 [0.08740234 0.76953125 0.1171875  0.02587891]
193 [0.09326172 0.78759766 0.10400391 0.01513672]
194 [0.10351562 0.76611328 0.10595703 0.02441406]
195 [0.09960938 0.77929688 0.09716797 0.02392578]
196 [0.09472656 0.77685547 0.10058594 0.02783203]
197 [0.09472656 0.77148438 0.10888672 0.02490234]
198 [0.09863281 0.77685547 0.09765625 0.02685547]
199 [0.10107422 0.77636719 0.10107422 0.02148438]
200 [0.09130859 0.77001953 0.11230469 0.02636719]
201 [0.1015625  0.76220703 0.11083984 0.02539062]
202 [0.09179688 0.77539062 0.11523438 0.01757812]
203 [0.09326172 0.77050781 0.10546875 0.03076172]
204 [0.10205078 0.76660156 0.10449219 0.02685547]
205 [0.09179688 0.76708984 0.11279297 0.02832031]
206 [0.09472656 0.76953125 0.11035156 0.02539062]
207 [0.10058594 0.76806641 0.10546875 0.02587891]
208 [0.08935547 0.77832031 0.10449219 0.02783203]
209 [0.09863281 0.77099609 0.10693359 0.0234375 ]
210 [0.08837891 0.77539062 0.11376953 0.02246094]
211 [0.09326172 0.75585938 0.125      0.02587891]
212 [0.09863281 0.77783203 0.09814453 0.02539062]
213 [0.10742188 0.765625   0.10742188 0.01953125]
214 [0.09130859 0.77294922 0.11328125 0.02246094]
215 [0.09521484 0.76953125 0.11474609 0.02050781]
216 [0.09375    0.76806641 0.11230469 0.02587891]
217 [0.09326172 0.77246094 0.11767578 0.01660156]
218 [0.08447266 0.77294922 0.11181641 0.03076172]
219 [0.08300781 0.78027344 0.11279297 0.02392578]
220 [0.08691406 0.77490234 0.10986328 0.02832031]
221 [0.10742188 0.77001953 0.09765625 0.02490234]
222 [0.08740234 0.78076172 0.11230469 0.01953125]
223 [0.0859375  0.78320312 0.109375   0.02148438]
224 [0.09667969 0.75341797 0.12597656 0.02392578]
225 [0.08837891 0.76416016 0.12109375 0.02636719]
226 [0.09179688 0.75878906 0.12548828 0.02392578]
227 [0.09912109 0.75341797 0.12158203 0.02587891]
228 [0.10205078 0.74707031 0.11962891 0.03125   ]
229 [0.09082031 0.76123047 0.1171875  0.03076172]
230 [0.09814453 0.75830078 0.11279297 0.03076172]
231 [0.10302734 0.75097656 0.11816406 0.02783203]
232 [0.08007812 0.76708984 0.12304688 0.02978516]
233 [0.09814453 0.76318359 0.1171875  0.02148438]
234 [0.09033203 0.76757812 0.11328125 0.02880859]
235 [0.09765625 0.76269531 0.11376953 0.02587891]
236 [0.08935547 0.7734375  0.11083984 0.02636719]
237 [0.09521484 0.75878906 0.12158203 0.02441406]
238 [0.09326172 0.75097656 0.12402344 0.03173828]
239 [0.08740234 0.76953125 0.11816406 0.02490234]
240 [0.11474609 0.73876953 0.12109375 0.02539062]
241 [0.09228516 0.77246094 0.11083984 0.02441406]
242 [0.10009766 0.74511719 0.12841797 0.02636719]
243 [0.07910156 0.75878906 0.13427734 0.02783203]
244 [0.07910156 0.77148438 0.12304688 0.02636719]
245 [0.1015625  0.76513672 0.10839844 0.02490234]
246 [0.09472656 0.75439453 0.12695312 0.02392578]
247 [0.10595703 0.74511719 0.11572266 0.03320312]
248 [0.10449219 0.75390625 0.11865234 0.02294922]
249 [0.08740234 0.77880859 0.10839844 0.02539062]
250 [0.08935547 0.75976562 0.12207031 0.02880859]
251 [0.08886719 0.765625   0.11083984 0.03466797]
252 [0.08984375 0.76855469 0.10986328 0.03173828]
253 [0.11035156 0.75195312 0.11328125 0.02441406]
254 [0.09179688 0.74414062 0.12939453 0.03466797]
255 [0.10253906 0.77050781 0.09960938 0.02734375]
256 [0.09228516 0.76416016 0.11132812 0.03222656]
257 [0.08447266 0.77441406 0.11621094 0.02490234]
258 [0.09667969 0.76953125 0.10986328 0.02392578]
259 [0.09082031 0.76123047 0.1171875  0.03076172]
260 [0.10058594 0.74804688 0.11523438 0.03613281]
261 [0.09423828 0.77880859 0.09912109 0.02783203]
262 [0.10205078 0.75683594 0.11230469 0.02880859]
263 [0.09082031 0.77587891 0.11181641 0.02148438]
264 [0.08740234 0.77636719 0.10644531 0.02978516]
265 [0.09863281 0.76367188 0.11523438 0.02246094]
266 [0.10449219 0.76660156 0.10351562 0.02539062]
```

```
[0.09716797 0.78320312 0.09521484 0.02441406]
[0.09570312 0.77001953 0.11083984 0.0234375 ]
[0.10351562 0.765625   0.11230469 0.01855469]
[0.09375    0.77294922 0.10693359 0.02636719]
[0.09765625 0.76269531 0.11523438 0.02441406]
[0.10644531 0.74560547 0.11816406 0.02978516]
[0.10009766 0.75585938 0.1171875  0.02685547]
[0.0859375  0.78027344 0.11132812 0.02246094]
[0.08544922 0.77392578 0.10986328 0.03076172]
[0.09716797 0.74609375 0.12988281 0.02685547]
[0.09716797 0.75927734 0.11230469 0.03125   ]
[0.11181641 0.75048828 0.11279297 0.02490234]
[0.09765625 0.75048828 0.12304688 0.02880859]
[0.10644531 0.75097656 0.11865234 0.02392578]
[0.09326172 0.75830078 0.12304688 0.02539062]
[0.07861328 0.75537109 0.13574219 0.03027344]
[0.09619141 0.75732422 0.12109375 0.02539062]
[0.09765625 0.74902344 0.13330078 0.02001953]
[0.09863281 0.75732422 0.12207031 0.02197266]
[0.10644531 0.74511719 0.12304688 0.02539062]
[0.09667969 0.74511719 0.13037109 0.02783203]
[0.08642578 0.76025391 0.12109375 0.03222656]
[0.09716797 0.75146484 0.12548828 0.02587891]
[0.09277344 0.74658203 0.13085938 0.02978516]
[0.09228516 0.74462891 0.125      0.03808594]
[0.08642578 0.73388672 0.14746094 0.03222656]
[0.09814453 0.73339844 0.13964844 0.02880859]
[0.08984375 0.74707031 0.13183594 0.03125   ]
[0.09423828 0.73535156 0.13769531 0.03271484]
[0.10351562 0.75292969 0.11914062 0.02441406]
[0.09960938 0.74316406 0.13232422 0.02490234]
[0.09814453 0.74316406 0.13037109 0.02832031]
[0.09326172 0.74951172 0.13085938 0.02636719]
[0.09179688 0.74902344 0.13085938 0.02832031]
[0.09765625 0.7578125  0.11279297 0.03173828]
Timer unit: 1e-07 s

Total time: 13.2161 s
File: C:/Users/usuario/qGAN/quantumGAN/performance_testing/performance_get_output_generator.py
Function: get_output_V1 at line 34

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    34                                           def get_output_V1():
    35       101       2839.0     28.1      0.0      for noise in batch_noise:
    36       100       2029.0     20.3      0.0          real_keys = {"00", "10", "01", "11"}
    37
    38       100      85545.0    855.5      0.1          quantum = QuantumRegister(sum(num_qubits), name="q")
    39       100     221056.0   2210.6      0.2          qc = QuantumCircuit(sum(num_qubits))
    40
    41       100     143365.0   1433.7      0.1          init_dist = qiskit.QuantumCircuit(sum(num_qubits))
    42       100       2823.0     28.2      0.0          assert noise.shape[0] == sum(num_qubits)
    43
    44       300       4768.0     15.9      0.0          for num_qubit in range(sum(num_qubits)):
    45       200     203976.0   1019.9      0.2              init_dist.ry(noise[num_qubit], num_qubit)
    46
    47       100       4094.0     40.9      0.0          params = cast(np.ndarray, parameter_values)
    48
    49       100   11231048.0 112310.5      8.5          qc.append(construct_circuit(params), quantum)
    50       100    3236878.0  32368.8      2.4          final_circuit = qc.compose(init_dist, front=True)
    51       100     319559.0   3195.6      0.2          final_circuit.measure_all()
    52
    53       100    3929383.0  39293.8      3.0          simulator_1 = qiskit.Aer.get_backend("aer_simulator")
    54       100  103884084.0 1038840.8     78.6          final_circuit = qiskit.transpile(final_circuit, simulator_1)
    55       100    8033391.0  80333.9      6.1          result = simulator_1.run(final_circuit, shots=shots).result()
    56       100     184560.0   1845.6      0.1          counts = result.get_counts(final_circuit)
    57
    58       100       1327.0     13.3      0.0          try:
    59       100      23239.0    232.4      0.0              pixels = np.array([counts["00"], counts["10"], counts["01
    "], counts["11"]])
    60
    61                                                   except KeyError:
    62                                                       # dealing with the keys that qiskit doesn't include in the
    63                                                       # dictionary because they don't get any measurements
    64
    65                                                       keys = counts.keys()
    66                                                       missing_keys = real_keys.difference(keys)
    67                                                       # we use sets to get the missing keys
    68                                                       for key_missing in missing_keys:
    69                                                           counts[key_missing] = 0
    70
    71                                                       pixels = np.array([counts["00"], counts["10"], counts["01
    "], counts["11"]])
    72
    73       100      29573.0    295.7      0.0          pixels = pixels / shots
    74       100     617900.0   6179.0      0.5          print(pixels)
    75
Total time: 12.5385 s
File: C:/Users/usuario/qGAN/quantumGAN/performance_testing/performance_get_output_generator.py
```

File - performance_get_output_generator

```
Function: get_output_V2 at line 77

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    77                                           def get_output_V2():
    78         1       3181.0   3181.0      0.0      simulator = qiskit.Aer.get_backend("aer_simulator")
    79       101       2660.0     26.3      0.0      for noise in batch_noise:
    80       100       2076.0     20.8      0.0          real_keys = {"00", "10", "01", "11"}
    81
    82       100      78460.0    784.6      0.1          quantum = QuantumRegister(sum(num_qubits), name="q")
    83       100     223361.0   2233.6      0.2          qc = QuantumCircuit(sum(num_qubits))
    84
    85       100     154854.0   1548.5      0.1          init_dist = qiskit.QuantumCircuit(sum(num_qubits))
    86       100       3814.0     38.1      0.0          assert noise.shape[0] == sum(num_qubits)
    87
    88       300       4776.0     15.9      0.0          for num_qubit in range(sum(num_qubits)):
    89       200     205933.0   1029.7      0.2              init_dist.ry(noise[num_qubit], num_qubit)
    90
    91       100       4055.0     40.5      0.0          params = cast(np.ndarray, parameter_values)
    92
    93       100   11126784.0 111267.8      8.9          qc.append(construct_circuit(params), quantum)
    94       100    3344374.0  33443.7      2.7          final_circuit = qc.compose(init_dist, front=True)
    95       100     340813.0   3408.1      0.3          final_circuit.measure_all()
    96
    97       100  100960154.0 1009601.5     80.5          final_circuit = qiskit.transpile(final_circuit, simulator)
    98       100    8051743.0  80517.4      6.4          result = simulator.run(final_circuit, shots=shots).result()
    99       100     187118.0   1871.2      0.1          counts = result.get_counts(final_circuit)
   100
   101       100       1415.0     14.2      0.0          try:
   102       100      22037.0    220.4      0.0              pixels = np.array([counts["00"], counts["10"], counts["01
    "], counts["11"]])
   103
   104                                                   except KeyError:
   105                                                       # dealing with the keys that qiskit doesn't include in the
   106                                                       # dictionary because they don't get any measurements
   107
   108                                                       keys = counts.keys()
   109                                                       missing_keys = real_keys.difference(keys)
   110                                                       # we use sets to get the missing keys
   111                                                       for key_missing in missing_keys:
   112                                                           counts[key_missing] = 0
   113
   114                                                       pixels = np.array([counts["00"], counts["10"], counts["01
    "], counts["11"]])
   115
   116       100      29790.0    297.9      0.0          pixels = pixels / shots
   117       100     637420.0   6374.2      0.5          print(pixels)

Total time: 14.6814 s
File: C:/Users/usuario/qGAN/quantumGAN/performance_testing/performance_get_output_generator.py
Function: get_output_V3 at line 119

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
   119                                           def get_output_V3():
   120         1       2683.0   2683.0      0.0      simulator = qiskit.Aer.get_backend("aer_simulator")
   121         1         16.0     16.0      0.0      real_keys = {"00", "10", "01", "11"}
   122
   123         1        654.0    654.0      0.0      quantum = QuantumRegister(sum(num_qubits), name="q")
   124         1       1617.0   1617.0      0.0      init_dist = qiskit.QuantumCircuit(sum(num_qubits))
   125
   126       101       2972.0     29.4      0.0      for noise in batch_noise:
   127       100       3931.0     39.3      0.0          assert noise.shape[0] == sum(num_qubits)
   128
   129       300       5306.0     17.7      0.0          for num_qubit in range(sum(num_qubits)):
   130       200     249137.0   1245.7      0.2              init_dist.ry(noise[num_qubit], num_qubit)
   131
   132       100       4341.0     43.4      0.0          params = cast(np.ndarray, parameter_values)
   133
   134       100     256200.0   2562.0      0.2          qc = QuantumCircuit(sum(num_qubits))
   135       100   11267432.0 112674.3      7.7          qc.append(construct_circuit(params), quantum)
   136       100    8327655.0  83276.6      5.7          final_circuit = qc.compose(init_dist, front=True)
   137       100     366109.0   3661.1      0.2          final_circuit.measure_all()
   138
   139       100  117567788.0 1175677.9     80.1          final_circuit = qiskit.transpile(final_circuit, simulator)
   140       100    7894014.0  78940.1      5.4          result = simulator.run(final_circuit, shots=shots).result()
   141       100     195841.0   1958.4      0.1          counts = result.get_counts(final_circuit)
   142
   143       100       1459.0     14.6      0.0          try:
   144       100      24636.0    246.4      0.0              pixels = np.array([counts["00"], counts["10"], counts["01
    "], counts["11"]])
   145
   146                                                   except KeyError:
   147                                                       # dealing with the keys that qiskit doesn't include in the
   148                                                       # dictionary because they don't get any measurements
   149
   150                                                       keys = counts.keys()
   151                                                       missing_keys = real_keys.difference(keys)
   152                                                       # we use sets to get the missing keys
```

File - performance_get_output_generator

```
440    153                                                      for key_missing in missing_keys:
441    154                                                          counts[key_missing] = 0
442    155
443    156                                                      pixels = np.array([counts["00"], counts["10"], counts["01
    "], counts["11"]])
444    157
445    158       100      32795.0    327.9      0.0         pixels = pixels / shots
446    159       100     608916.0   6089.2      0.4         print(pixels)
447
448
449 Process finished with exit code 0
450
```