High Dynamic Range (HDR) Imaging

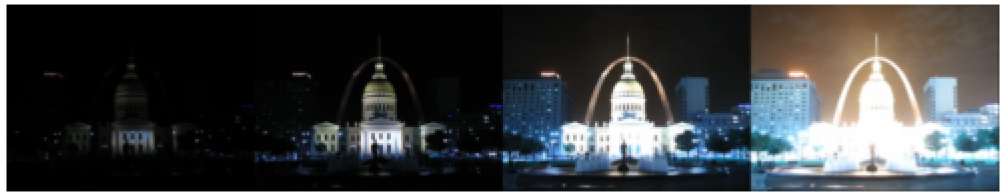
In [1]:
```
#1: The Dynamic Range od Images is Limited to 8-bits (0-255) Per Channel
#2: Very Bright Pixels Saturated to 255
#3: Very Dark Pixels Clip to 0
```

1: Capture Muliple Exposures

In [2]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
sampleImage = cv2.imread("hdrsample.jpg", cv2.IMREAD_COLOR)
plt.imshow(sampleImage)
plt.axis("off")
```

Out[3]:
```
(-0.5, 1865.5, 349.5, -0.5)
```

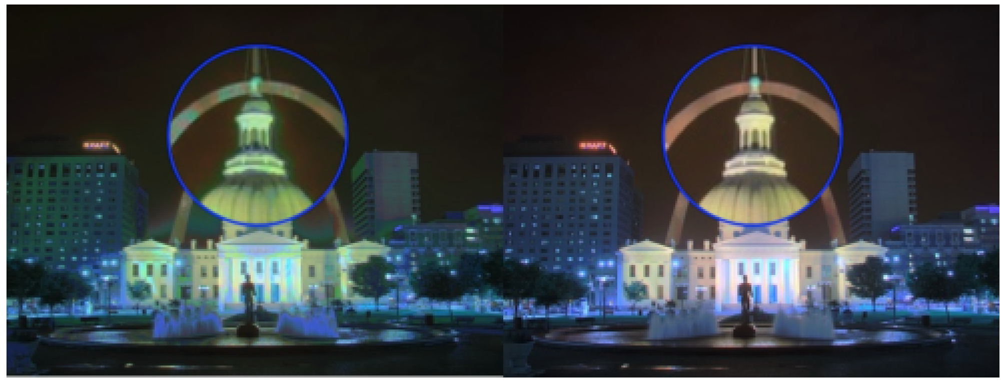


In [18]:
```
def readImagesAndTimes():
    # List of file names
    filenames = ["img_0.033.jpg", "img_0.25.jpg", "img_2.5.jpg", "img_15.jpg"]

    # List of exposure times
    times = np.array([1 / 30.0, 0.25, 2.5, 15.0], dtype=np.float32)

    # Read images
    images = []
    for filename in filenames:
        img = cv2.imread(filename)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        images.append(img)

    return images, times
```

2: Align Images

In [19]:
```
alignImage = cv2.imread("alignsample.jpg", cv2.IMREAD_COLOR)
plt.imshow(alignImage)
plt.axis("off")
```

Out[19]:
```
(-0.5, 999.5, 375.5, -0.5)
```



In [21]:
```
# Read images and exposure times
images, times = readImagesAndTimes()

# Align Images
alignMTB = cv2.createAlignMTB()
alignMTB.process(images, images)
```

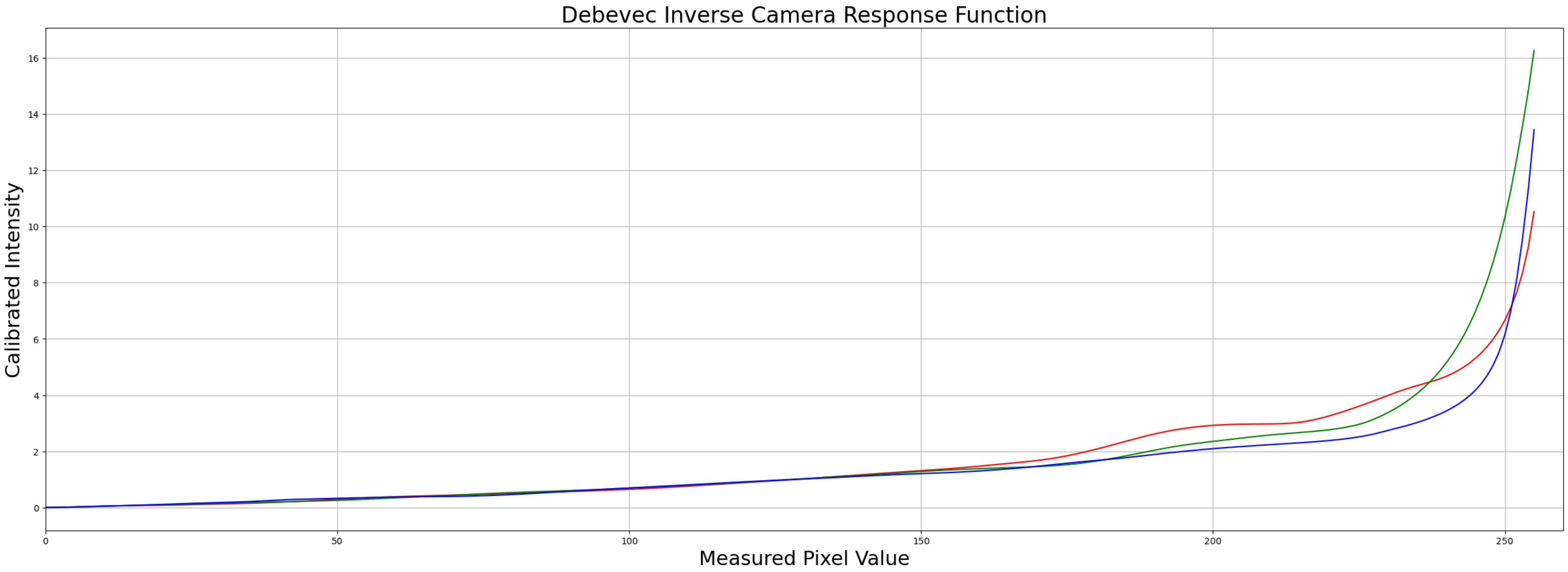
3: Estimate Camera Response Function

In [22]:

```
# Find Camera Response Function (CRF)
calibrateDebevec = cv2.createCalibrateDebevec()
responseDebevec = calibrateDebevec.process(images, times)

# Plot CRF
x = np.arange(256, dtype=np.uint8)
y = np.squeeze(responseDebevec)

ax = plt.figure(figsize=(30, 10))
plt.title("Debevec Inverse Camera Response Function", fontsize=24)
plt.xlabel("Measured Pixel Value", fontsize=22)
plt.ylabel("Calibrated Intensity", fontsize=22)
plt.xlim([0, 260])
plt.grid()
plt.plot(x, y[:, 0], "r", x, y[:, 1], "g", x, y[:, 2], "b")
```

Out[22]:

```
[<matplotlib.lines.Line2D at 0x2410d452a90>,
 <matplotlib.lines.Line2D at 0x2410d452a60>,
 <matplotlib.lines.Line2D at 0x2410d452ac0>]
```



4: Merge Exposure into an HDR Image

In [23]:

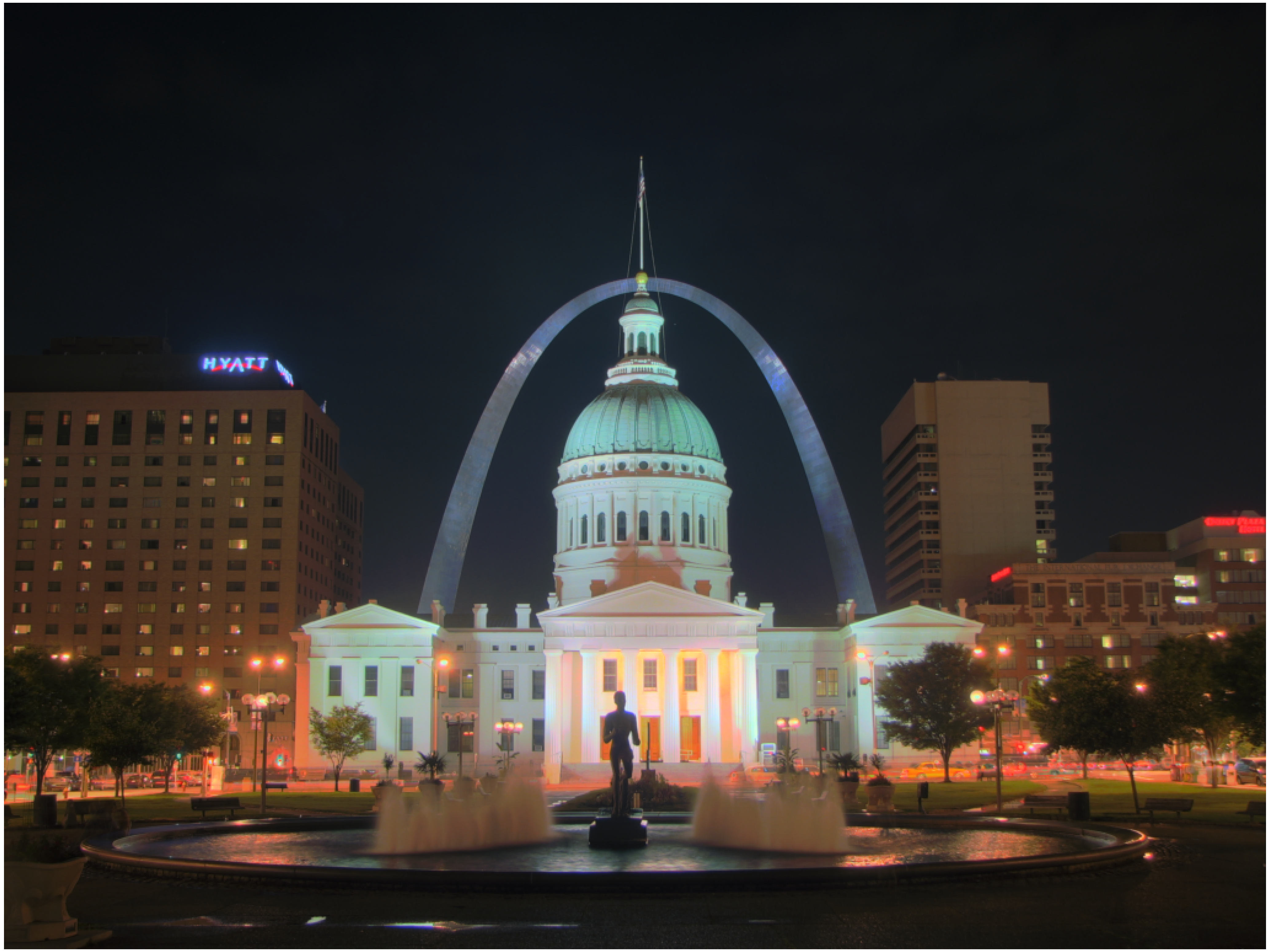
```
# Merge images into an HDR linear image
mergeDebevec = cv2.createMergeDebevec()
hdrDebevec = mergeDebevec.process(images, times, responseDebevec)
```

5: Tonemapping

In [24]:

```
# Tonemap using Drago's method to obtain 24-bit color image
tonemapDrago = cv2.createTonemapDrago(1.0, 0.7)
ldrDrago = tonemapDrago.process(hdrDebevec)
ldrDrago = 3 * ldrDrago

plt.figure(figsize=(20, 10));plt.imshow(np.clip(ldrDrago, 0, 1));plt.axis("off")

cv2.imwrite("ldr-Drago.png", ldrDrago * 255)
print("saved ldr-Drago.png")
```

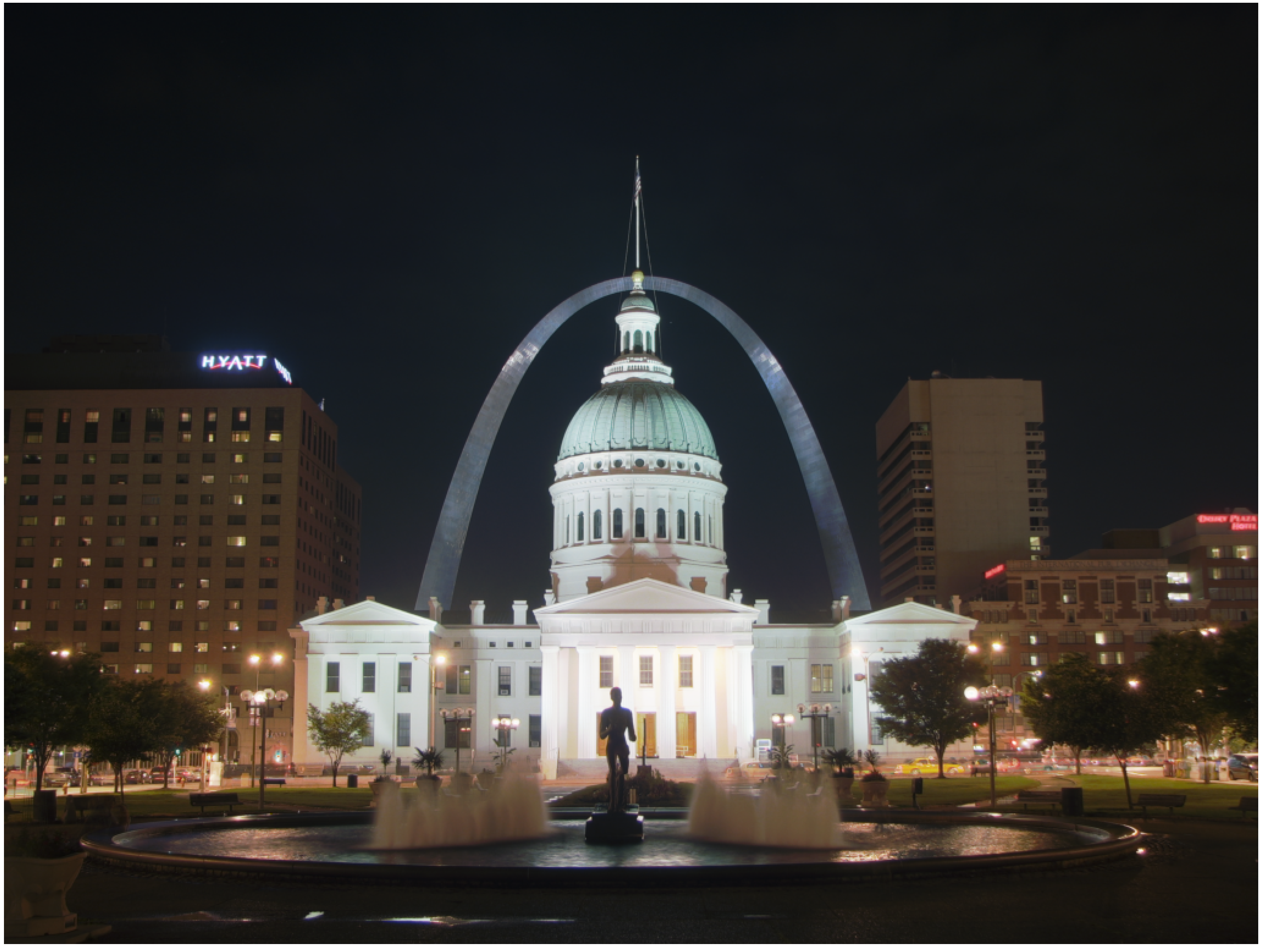
```
saved ldr-Drago.png
```



In [25]:

```
# Tonemap using Reinhard's method to obtain 24-bit color image
print("Tonemaping using Reinhard's method ... ")
tonemapReinhard = cv2.createTonemapReinhard(1.5, 0, 0, 0)
ldrReinhard = tonemapReinhard.process(hdrDebevec)

plt.figure(figsize=(20, 10));plt.imshow(np.clip(ldrReinhard, 0, 1));plt.axis("off")

cv2.imwrite("ldr-Reinhard.png", ldrReinhard * 255)
print("saved ldr-Reinhard.png")
```

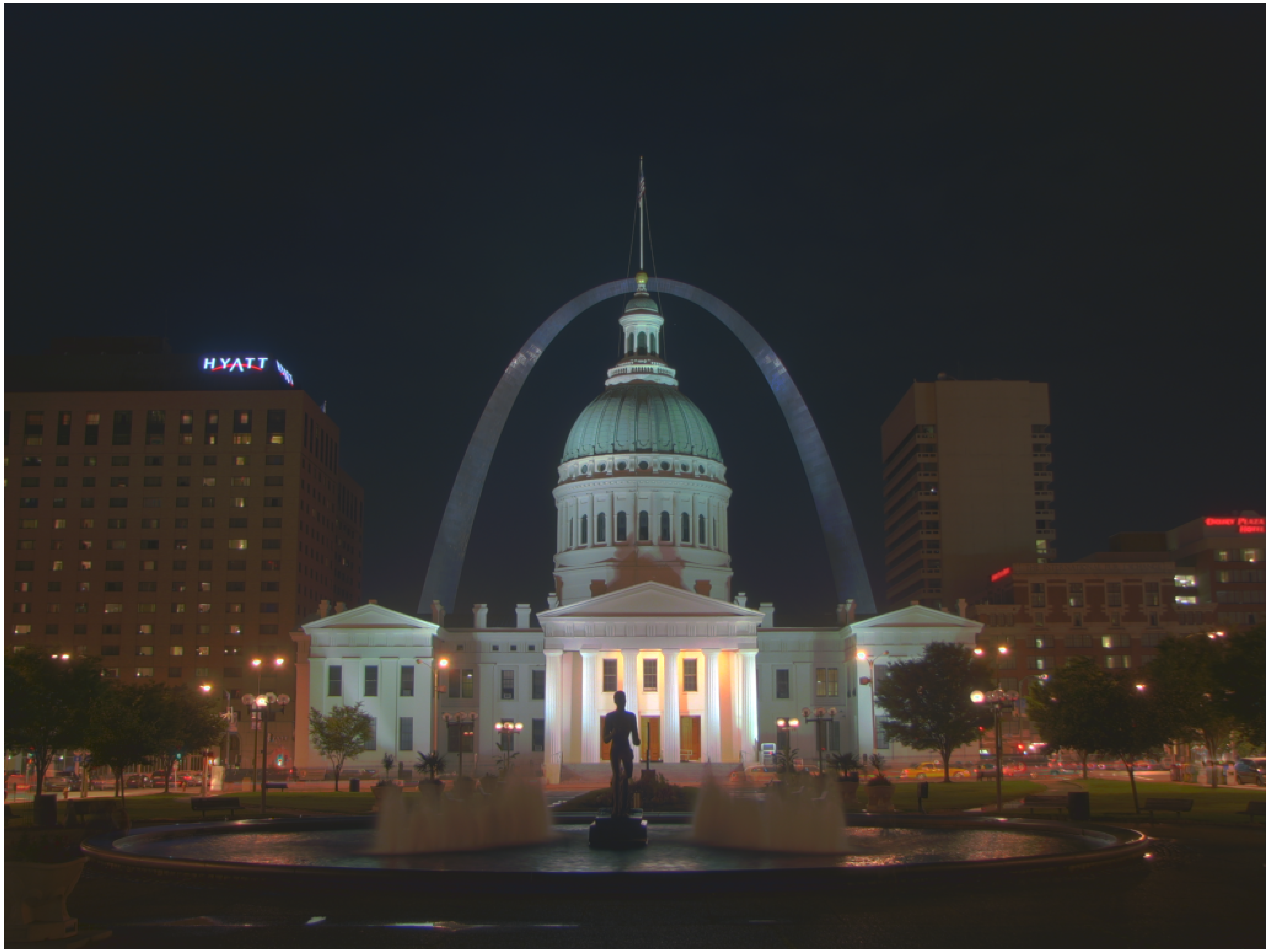
```
Tonemaping using Reinhard's method ...
saved ldr-Reinhard.png
```



In [26]:

```
# Tonemap using Mantiuk's method to obtain 24-bit color image
print("Tonemaping using Mantiuk's method ... ")
tonemapMantiuk = cv2.createTonemapMantiuk(2.2, 0.85, 1.2)
ldrMantiuk = tonemapMantiuk.process(hdrDebevec)
ldrMantiuk = 3 * ldrMantiuk

plt.figure(figsize=(20, 10));plt.imshow(np.clip(ldrMantiuk, 0, 1));plt.axis("off")

cv2.imwrite("ldr-Mantiuk.png", ldrMantiuk * 255)
print("saved ldr-Mantiuk.png")
```

```
Tonemaping using Mantiuk's method ...
saved ldr-Mantiuk.png
```


HYATT