

When Synthetic Data makes sense in Computer Vision

Real-world edge cases are hard.

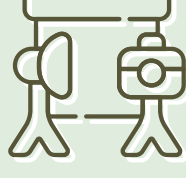
Here's how synthetics fill the gaps.

Use
synthetic
data when
you need...



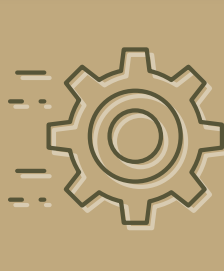
Rare Edge Cases

near-misses, unusual object interactions, uncommon scenarios



Controlled Coverage

lighting, backgrounds, occlusion levels, camera angles



Fast Iteration

bootstrapping a model before full-scale collection



Privacy & Compliance Constraints

sensitive environments where real capture is restricted



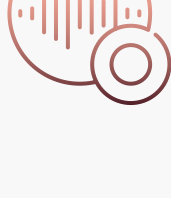
Class Balance Support

boosting underrepresented classes for training stability

Watch-outs

(synthetic can backfire if...)

Visuals don't match
sensor reality (noise,
blur, lens artifacts)



Scene physics look

"off" (shadows,
reflections, contact)



Domain gap isn't
measured (you
train on synthetic,
deploy in real)



You validate only on

synthetic (leads to
false confidence)



Best-practice approach

**Define the
target domain**
(where the
model will run)

**Generate
synthetic
variations**
aligned to that
domain

**Blend with
real data**
(don't replace it)

**Validate on
real holdout**
(must-have)

**Iterate using
error analysis**
(use failures to
guide next
synthetic batch)

Should you use synthetic data?

Is the scenario **rare or risky** to capture?

Yes → Synthetic helps

Do you need **controlled variation** quickly?

Yes → Synthetic helps

Is your model failing due to **domain shift**?

**Yes → Use real + domain-matched
synthetic**

Can you evaluate reliably on **real
holdout data**?

No → Don't rely on synthetic yet

Shaip helps CV teams plan
the right mix of
real + synthetic data,
with measurable QA and
deployment-focused validation.