ДОДАТОК А
(довідковий)

«ЛІСТИНГ ПРОГРАМНОГО КОДУ»

```
	#ifdef _MSC_VER
	#pragma warning (push)
	#pragma warning (disable : 4883)
	#endif
	PRAGMA_DISABLE_DEPRECATION_WARNINGS
	PRAGMA_DISABLE_OPTIMIZATION
	UShadows_C__pf2937682852::UShadows_C__pf2937682852(const  FObjectInitializer&
ObjectInitializer) : Super(ObjectInitializer)
	{

	bpv__Amb__pf = nullptr;
	bpv__AO__pf = nullptr;
	bpv__AOSamles__pf = nullptr;
	bpv__AOText__pf = nullptr;
	bpv__Camera1__pf = nullptr;
	bpv__Camera2__pf = nullptr;
	bpv__Camera3__pf = nullptr;
	bpv__Direct__pf = nullptr;
	bpv__DR__pf = nullptr;
	bpv__GI__pf = nullptr;
	bpv__GIRays__pf = nullptr;
	bpv__GIraystext__pf = nullptr;
	bpv__Glob__pf = nullptr;
	bpv__Ref__pf = nullptr;
	bpv__Refle__pf = nullptr;
	bpv__RefRays__pf = nullptr;
	bpv__RefRaysText__pf = nullptr;
	bpv__RefRo__pf = nullptr;
	bpv__REROGHTExt__pf = nullptr;
	bpv__TR__pf = nullptr;
	bpv__Trans__pf = nullptr;
	bpv__TransRays__pf = nullptr;
	bpv__TransRaysText__pf = nullptr;
	bpv__TransRoghText__pf = nullptr;
	bpv__TrasRogh__pf = nullptr;
	bpv__Volume__pf = nullptr;
	bpv__Pawn1__pf = nullptr;
	bpv__Pawn2__pf = nullptr;
	bpv__Pawn3__pf = nullptr;
	bHasScriptImplementedTick = false;
	bHasScriptImplementedPaint = false;
	}
	PRAGMA_ENABLE_OPTIMIZATION
	void     UShadows_C__pf2937682852::PostLoadSubobjects(FObjectInstancingGraph*
OuterInstanceGraph)
	{
	Super::PostLoadSubobjects(OuterInstanceGraph);
	}
	PRAGMA_DISABLE_OPTIMIZATION
	void
UShadows_C__pf2937682852::__CustomDynamicClassInitialization(UDynamicClass*
InDynamicClass)
	{
	ensure(0 == InDynamicClass->ReferencedConvertedFields.Num());
	ensure(0 == InDynamicClass->MiscConvertedSubobjects.Num());
	ensure(0 == InDynamicClass->DynamicBindingObjects.Num());
	ensure(0 == InDynamicClass->ComponentTemplates.Num());
	ensure(0 == InDynamicClass->Timelines.Num());
	ensure(0 == InDynamicClass->ComponentClassOverrides.Num());
	ensure(nullptr == InDynamicClass->AnimClassImplementation);
	InDynamicClass->AssembleReferenceTokenStream();
	FConvertedBlueprintsDependencies::FillUsedAssetsInDynamicClass(InDynamicClas
s, &__StaticDependencies_DirectlyUsedAssets);
```

```
	auto __Local__0 = NewObject<UComponentDelegateBinding>(InDynamicClass,
UComponentDelegateBinding::StaticClass(), TEXT("ComponentDelegateBinding_1"),
(EObjectFlags)0x00000000);
	InDynamicClass->DynamicBindingObjects.Add(__Local__0);
	auto __Local__1 = NewObject<UWidgetTree>(InDynamicClass,
UWidgetTree::StaticClass(), TEXT("WidgetTree"), (EObjectFlags)0x00000009);
	InDynamicClass->MiscConvertedSubobjects.Add(__Local__1);
	__Local__0->ComponentDelegateBindings =
TArray<FBlueprintComponentDelegateBinding> ();
	__Local__0->ComponentDelegateBindings.AddUninitialized(14);
	FBlueprintComponentDelegateBinding::StaticStruct()-
>InitializeStruct(__Local__0->ComponentDelegateBindings.GetData(), 14);
	auto& __Local__2 = __Local__0->ComponentDelegateBindings[0];
	PRAGMA_ENABLE_OPTIMIZATION
	void UShadows_C__pf2937682852::GetSlotNames(TArray<FName>& SlotNames) const
	{
	TArray<FName>  __Local__98;
	SlotNames.Append(__Local__98);
	}
	void UShadows_C__pf2937682852::InitializeNativeClassData()
	{
	TArray<UWidgetAnimation*>  __Local__99;
	TArray<FDelegateRuntimeBinding>  __Local__100;
	UWidgetBlueprintGeneratedClass::InitializeWidgetStatic(this, GetClass(),
false, true,
CastChecked<UWidgetTree>(CastChecked<UDynamicClass>(UShadows_C__pf2937682852::Stat
icClass())->MiscConvertedSubobjects[0]), __Local__99, __Local__100);
	}
	void UShadows_C__pf2937682852::PreSave(const class ITargetPlatform*
TargetPlatform)
	{
	Super::PreSave(TargetPlatform);
	TArray<FName> LocalNamedSlots;
	GetSlotNames(LocalNamedSlots);
	RemoveObsoleteBindings(LocalNamedSlots);
	}
	void UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_0(int32
bpp__EntryPoint__pf)
	{
	float bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_3__pf{};
	int32 bpfv__CallFunc_FTrunc_ReturnValue_3__pf{};
	int32 bpfv__CallFunc_Add_IntInt_ReturnValue_4__pf{};
	FString bpfv__CallFunc_Conv_IntToString_ReturnValue_3__pf{};
	FString bpfv__CallFunc_MakeLiteralString_ReturnValue_5__pf{};
	FString bpfv__CallFunc_Concat_StrStr_ReturnValue_5__pf{};
	FText bpfv__CallFunc_Conv_StringToText_ReturnValue_5__pf{};
	check(bpp__EntryPoint__pf == 58);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_3__pf =
UKismetMathLibrary::Multiply_FloatFloat(b01__K2Node_ComponentBoundEvent_Value__pf,
49.000000);
	bpfv__CallFunc_FTrunc_ReturnValue_3__pf =
UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_3__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_4__pf =
UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue_3__pf, 1);
	bpfv__CallFunc_Conv_IntToString_ReturnValue_3__pf =
UKismetStringLibrary::Conv_IntToString(bpfv__CallFunc_Add_IntInt_ReturnValue_4__pf
);
	bpfv__CallFunc_MakeLiteralString_ReturnValue_5__pf =
UKismetSystemLibrary::MakeLiteralString(FString(TEXT("AO Samples: ")));
	bpfv__CallFunc_Concat_StrStr_ReturnValue_5__pf =
UKismetStringLibrary::Concat_StrStr(bpfv__CallFunc_MakeLiteralString_ReturnValue_5
__pf, bpfv__CallFunc_Conv_IntToString_ReturnValue_3__pf);
```

```
	bpfv__CallFunc_Conv_StringToText_ReturnValue_5__pf = UKismetTextLibrary::Conv_StringToText(bpfv__CallFunc_Concat_StrStr_ReturnValue_5__pf);
	if(::IsValid(bpv__AOText__pf))
	{
		bpv__AOText__pf->SetText(bpfv__CallFunc_Conv_StringToText_ReturnValue_5__pf);
	}
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_3__pf = UKismetMathLibrary::Multiply_FloatFloat(b0l__K2Node_ComponentBoundEvent_Value__pf, 49.000000);
	bpfv__CallFunc_FTrunc_ReturnValue_3__pf = UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_3__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_4__pf = UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue_3__pf, 1);
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.RayTracingAOSamplesPerPixel = bpfv__CallFunc_Add_IntInt_ReturnValue_4__pf;
	}
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.bOverride_RayTracingAOSamplesPerPixel = true;
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_1(int32 bpp__EntryPoint__pf)
	{
	float bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_1__pf{};
	int32 bpfv__CallFunc_FTrunc_ReturnValue_1__pf{};
	int32 bpfv__CallFunc_Add_IntInt_ReturnValue_2__pf{};
	FString bpfv__CallFunc_Conv_IntToString_ReturnValue_1__pf{};
	FString bpfv__CallFunc_MakeLiteralString_ReturnValue_4__pf{};
	FString bpfv__CallFunc_Concat_StrStr_ReturnValue_4__pf{};
	FText bpfv__CallFunc_Conv_StringToText_ReturnValue_4__pf{};
	check(bpp__EntryPoint__pf == 49);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_1__pf = UKismetMathLibrary::Multiply_FloatFloat(b0l__K2Node_ComponentBoundEvent_Value_3__pf, 49.000000);
	bpfv__CallFunc_FTrunc_ReturnValue_1__pf = UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_1__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_2__pf = UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue_1__pf, 1);
	bpfv__CallFunc_Conv_IntToString_ReturnValue_1__pf = UKismetStringLibrary::Conv_IntToString(bpfv__CallFunc_Add_IntInt_ReturnValue_2__pf);
	bpfv__CallFunc_MakeLiteralString_ReturnValue_4__pf = UKismetSystemLibrary::MakeLiteralString(FString(TEXT("GI Samples: ")));
	bpfv__CallFunc_Concat_StrStr_ReturnValue_4__pf = UKismetStringLibrary::Concat_StrStr(bpfv__CallFunc_MakeLiteralString_ReturnValue_4__pf, bpfv__CallFunc_Conv_IntToString_ReturnValue_1__pf);
	bpfv__CallFunc_Conv_StringToText_ReturnValue_4__pf = UKismetTextLibrary::Conv_StringToText(bpfv__CallFunc_Concat_StrStr_ReturnValue_4__pf);
	if(::IsValid(bpv__GIraystext__pf))
	{
		bpv__GIraystext__pf->SetText(bpfv__CallFunc_Conv_StringToText_ReturnValue_4__pf);
	}
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_1__pf = UKismetMathLibrary::Multiply_FloatFloat(b0l__K2Node_ComponentBoundEvent_Value_3__pf, 49.000000);
```

```
	bpfv__CallFunc_FTrunc_ReturnValue_1__pf = UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_1__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_2__pf = UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue_1__pf, 1);
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.RayTracingGISamplesPerPixel = bpfv__CallFunc_Add_IntInt_ReturnValue_2__pf;
	}
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.bOverride_RayTracingGISamplesPerPixel = true;
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_2(int32 bpp__EntryPoint__pf)
	{
	float bpfv__CallFunc_FClamp_ReturnValue_1__pf{};
	FString bpfv__CallFunc_Conv_FloatToString_ReturnValue_1__pf{};
	FString bpfv__CallFunc_MakeLiteralString_ReturnValue_2__pf{};
	FString bpfv__CallFunc_Concat_StrStr_ReturnValue_2__pf{};
	FText bpfv__CallFunc_Conv_StringToText_ReturnValue_2__pf{};
	check(bpp__EntryPoint__pf == 52);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_FClamp_ReturnValue_1__pf = UKismetMathLibrary::FClamp(b01__K2Node_ComponentBoundEvent_Value_2__pf,  0.010000, 1.000000);
	bpfv__CallFunc_Conv_FloatToString_ReturnValue_1__pf = UKismetStringLibrary::Conv_FloatToString(bpfv__CallFunc_FClamp_ReturnValue_1__pf);
	bpfv__CallFunc_MakeLiteralString_ReturnValue_2__pf = UKismetSystemLibrary::MakeLiteralString(FString(TEXT("Reflection max roghness: ")));
	bpfv__CallFunc_Concat_StrStr_ReturnValue_2__pf = UKismetStringLibrary::Concat_StrStr(bpfv__CallFunc_MakeLiteralString_ReturnValue_2__pf, bpfv__CallFunc_Conv_FloatToString_ReturnValue_1__pf);
	bpfv__CallFunc_Conv_StringToText_ReturnValue_2__pf = UKismetTextLibrary::Conv_StringToText(bpfv__CallFunc_Concat_StrStr_ReturnValue_2__pf);
	if(::IsValid(bpv__REROGHTExt__pf))
	{
		bpv__REROGHTExt__pf->SetText(bpfv__CallFunc_Conv_StringToText_ReturnValue_2__pf);
	}
	bpfv__CallFunc_FClamp_ReturnValue_1__pf = UKismetMathLibrary::FClamp(b01__K2Node_ComponentBoundEvent_Value_2__pf,  0.010000, 1.000000);
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.RayTracingReflectionsMaxRoughness = bpfv__CallFunc_FClamp_ReturnValue_1__pf;
	}
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.bOverride_RayTracingReflectionsMaxRoughness = true;
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_3(int32 bpp__EntryPoint__pf)
	{
	float bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_2__pf{};
	int32 bpfv__CallFunc_FTrunc_ReturnValue_2__pf{};
```

```
	int32 bpfv__CallFunc_Add_IntInt_ReturnValue_3__pf{};
	FString bpfv__CallFunc_Conv_IntToString_ReturnValue_2__pf{};
	FString bpfv__CallFunc_MakeLiteralString_ReturnValue_3__pf{};
	FString bpfv__CallFunc_Concat_StrStr_ReturnValue_3__pf{};
	FText bpfv__CallFunc_Conv_StringToText_ReturnValue_3__pf{};
	check(bpp__EntryPoint__pf == 55);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_2__pf                    =
UKismetMathLibrary::Multiply_FloatFloat(b01__K2Node_ComponentBoundEvent_Value_1__p
f, 49.000000);
	bpfv__CallFunc_FTrunc_ReturnValue_2__pf                                 =
UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_2__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_3__pf                             =
UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue_2__pf, 1);
	bpfv__CallFunc_Conv_IntToString_ReturnValue_2__pf                       =
UKismetStringLibrary::Conv_IntToString(bpfv__CallFunc_Add_IntInt_ReturnValue_3__pf
);
	bpfv__CallFunc_MakeLiteralString_ReturnValue_3__pf                      =
UKismetSystemLibrary::MakeLiteralString(FString(TEXT("Reflection  rays: ")));
	bpfv__CallFunc_Concat_StrStr_ReturnValue_3__pf                          =
UKismetStringLibrary::Concat_StrStr(bpfv__CallFunc_MakeLiteralString_ReturnValue_3
__pf, bpfv__CallFunc_Conv_IntToString_ReturnValue_2__pf);
	bpfv__CallFunc_Conv_StringToText_ReturnValue_3__pf                      =
UKismetTextLibrary::Conv_StringToText(bpfv__CallFunc_Concat_StrStr_ReturnValue_3__
pf);
	if(::IsValid(bpv__RefRaysText__pf))
	{
		bpv__RefRaysText__pf-
>SetText(bpfv__CallFunc_Conv_StringToText_ReturnValue_3__pf);
	}
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_2__pf                    =
UKismetMathLibrary::Multiply_FloatFloat(b01__K2Node_ComponentBoundEvent_Value_1__p
f, 49.000000);
	bpfv__CallFunc_FTrunc_ReturnValue_2__pf                                 =
UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue_2__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_3__pf                             =
UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue_2__pf, 1);
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.RayTracingReflectionsMaxBounces           =
bpfv__CallFunc_Add_IntInt_ReturnValue_3__pf;
	}
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.bOverride_RayTracingReflectionsMaxBounces   =
true;
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_4(int32
bpp__EntryPoint__pf)
	{
	float bpfv__CallFunc_Multiply_FloatFloat_ReturnValue__pf{};
	int32 bpfv__CallFunc_FTrunc_ReturnValue__pf{};
	int32 bpfv__CallFunc_Add_IntInt_ReturnValue_1__pf{};
	FString bpfv__CallFunc_Conv_IntToString_ReturnValue__pf{};
	FString bpfv__CallFunc_MakeLiteralString_ReturnValue_1__pf{};
	FString bpfv__CallFunc_Concat_StrStr_ReturnValue_1__pf{};
	FText bpfv__CallFunc_Conv_StringToText_ReturnValue_1__pf{};
	check(bpp__EntryPoint__pf == 46);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue__pf                      =
UKismetMathLibrary::Multiply_FloatFloat(b01__K2Node_ComponentBoundEvent_Value_4__p
f, 49.000000);
```

```
	bpfv__CallFunc_FTrunc_ReturnValue__pf                                    =
UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_1__pf                              =
UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue__pf, 1);
	bpfv__CallFunc_Conv_IntToString_ReturnValue__pf                          =
UKismetStringLibrary::Conv_IntToString(bpfv__CallFunc_Add_IntInt_ReturnValue_1__pf
);
	bpfv__CallFunc_MakeLiteralString_ReturnValue_1__pf                       =
UKismetSystemLibrary::MakeLiteralString(FString(TEXT("Translucent  rays: ")));
	bpfv__CallFunc_Concat_StrStr_ReturnValue_1__pf                           =
UKismetStringLibrary::Concat_StrStr(bpfv__CallFunc_MakeLiteralString_ReturnValue_1
__pf, bpfv__CallFunc_Conv_IntToString_ReturnValue__pf);
	bpfv__CallFunc_Conv_StringToText_ReturnValue_1__pf                       =
UKismetTextLibrary::Conv_StringToText(bpfv__CallFunc_Concat_StrStr_ReturnValue_1__
pf);
	if(::IsValid(bpv__TransRaysText__pf))
	{
		bpv__TransRaysText__pf-
>SetText(bpfv__CallFunc_Conv_StringToText_ReturnValue_1__pf);
	}
	bpfv__CallFunc_Multiply_FloatFloat_ReturnValue__pf                       =
UKismetMathLibrary::Multiply_FloatFloat(b01__K2Node_ComponentBoundEvent_Value_4__p
f, 49.000000);
	bpfv__CallFunc_FTrunc_ReturnValue__pf                                    =
UKismetMathLibrary::FTrunc(bpfv__CallFunc_Multiply_FloatFloat_ReturnValue__pf);
	bpfv__CallFunc_Add_IntInt_ReturnValue_1__pf                              =
UKismetMathLibrary::Add_IntInt(bpfv__CallFunc_FTrunc_ReturnValue__pf, 1);
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.RayTracingTranslucencyRefractionRays       =
bpfv__CallFunc_Add_IntInt_ReturnValue_1__pf;
	}
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf-
>Settings.bOverride_RayTracingTranslucencyRefractionRays = true;
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_5(int32
bpp__EntryPoint__pf)
	{
	float bpfv__CallFunc_FClamp_ReturnValue__pf{};
	FString bpfv__CallFunc_Conv_FloatToString_ReturnValue__pf{};
	FString bpfv__CallFunc_MakeLiteralString_ReturnValue__pf{};
	FString bpfv__CallFunc_Concat_StrStr_ReturnValue__pf{};
	FText bpfv__CallFunc_Conv_StringToText_ReturnValue__pf{};
	check(bpp__EntryPoint__pf == 43);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_FClamp_ReturnValue__pf                                    =
UKismetMathLibrary::FClamp(b01__K2Node_ComponentBoundEvent_Value_5__pf,  0.010000,
1.000000);
	bpfv__CallFunc_Conv_FloatToString_ReturnValue__pf                        =
UKismetStringLibrary::Conv_FloatToString(bpfv__CallFunc_FClamp_ReturnValue__pf);
	bpfv__CallFunc_MakeLiteralString_ReturnValue__pf                         =
UKismetSystemLibrary::MakeLiteralString(FString(TEXT("Translucent  max  roghness:
")));
	bpfv__CallFunc_Concat_StrStr_ReturnValue__pf                             =
UKismetStringLibrary::Concat_StrStr(bpfv__CallFunc_MakeLiteralString_ReturnValue__
pf, bpfv__CallFunc_Conv_FloatToString_ReturnValue__pf);
	bpfv__CallFunc_Conv_StringToText_ReturnValue__pf                         =
UKismetTextLibrary::Conv_StringToText(bpfv__CallFunc_Concat_StrStr_ReturnValue__pf
);
	if(::IsValid(bpv__TransRoghText__pf))
```

```
	{
		bpv__TransRoghText__pf->SetText(bpfv__CallFunc_Conv_StringToText_ReturnValue__pf);
	}
	bpfv__CallFunc_FClamp_ReturnValue__pf = UKismetMathLibrary::FClamp(b0l__K2Node_ComponentBoundEvent_Value_5__pf,  0.010000, 1.000000);
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.RayTracingTranslucencyMaxRoughness = bpfv__CallFunc_FClamp_ReturnValue__pf;
	}
	if(::IsValid(bpv__Volume__pf))
	{
		bpv__Volume__pf->Settings.bOverride_RayTracingTranslucencyMaxRoughness = true;
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_6(int32 bpp__EntryPoint__pf)
	{
	bool bpfv__CallFunc_Not_PreBool_ReturnValue_1__pf{};
	int32 __CurrentState = bpp__EntryPoint__pf;
	do
	{
		switch( __CurrentState )
		{
		case 12:
			{
				if (!b0l__Temp_bool_Variable_11__pf)
				{
					__CurrentState = 13;
					break;
				}
			}
		case 13:
			{
				b0l__Temp_byte_Variable_12__pf = ETranslucencyType::RayTracing;
				b0l__Temp_bool_Variable_13__pf = b0l__Temp_bool_Variable_11__pf;
				b0l__Temp_byte_Variable_13__pf = ETranslucencyType::Raster;
				if(::IsValid(bpv__Volume__pf))
				{
					bpv__Volume__pf->Settings.TranslucencyType = TSwitchValue<bool , ETranslucencyType >(b0l__Temp_bool_Variable_13__pf, b0l__K2Node_Select_Default_11__pf, 2, TSwitchPair<bool , ETranslucencyType >(false, b0l__Temp_byte_Variable_13__pf), TSwitchPair<bool , ETranslucencyType >(true, b0l__Temp_byte_Variable_12__pf));
				}
				if(::IsValid(bpv__Volume__pf))
				{
					bpv__Volume__pf->Settings.bOverride_TranslucencyType = true;
				}
			}
		case 14:
			{
				b0l__Temp_text_Variable_2__pf = FTextStringHelper::CreateFromBuffer(
	TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\", \"0E9659964BE710F09F66EBA79DA5E484\", \"Ray tracing\")")    );
```

```
                        b01__Temp_text_Variable_3__pf                                   =
FTextStringHelper::CreateFromBuffer(
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"4C63B2E7440A85BB884217BCA157DBE2\", \"Raster\")")   );
                        b01__Temp_bool_Variable_2__pf                                   =
b01__Temp_bool_Variable_11__pf;
                        if(::IsValid(bpv__TR__pf))
                        {
                              bpv__TR__pf->SetText(TSwitchValue<bool     ,     FText
>(b01__Temp_bool_Variable_2__pf,        b01__K2Node_Select_Default_1__pf,        2,
TSwitchPair<bool  , FText >(false, b01__Temp_text_Variable_3__pf), TSwitchPair<bool
, FText >(true, b01__Temp_text_Variable_2__pf)));
                        }
                  }
            case 15:
                  {
                        b01__Temp_byte_Variable_3__pf = ESlateVisibility::Visible;
                        b01__Temp_byte_Variable_4__pf = ESlateVisibility::Hidden;
                        b01__Temp_bool_Variable_6__pf                                   =
b01__Temp_bool_Variable_11__pf;
                        if(::IsValid(bpv__TransRays__pf))
                        {
                              bpv__TransRays__pf->SetVisibility(TSwitchValue<bool
,                ESlateVisibility                >(b01__Temp_bool_Variable_6__pf,
b01__K2Node_Select_Default_2__pf, 2, TSwitchPair<bool , ESlateVisibility >(false,
b01__Temp_byte_Variable_4__pf),   TSwitchPair<bool   ,   ESlateVisibility   >(true,
b01__Temp_byte_Variable_3__pf)));
                        }
                        if(::IsValid(bpv__TrasRogh__pf))
                        {
                              bpv__TrasRogh__pf->SetVisibility(TSwitchValue<bool  ,
ESlateVisibility >(b01__Temp_bool_Variable_6__pf, b01__K2Node_Select_Default_2__pf,
2,  TSwitchPair<bool  ,  ESlateVisibility  >(false,  b01__Temp_byte_Variable_4__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_3__pf)));
                        }
                        if(::IsValid(bpv__TransRaysText__pf))
                        {
                              bpv__TransRaysText__pf-
>SetVisibility(TSwitchValue<bool                ,                ESlateVisibility
>(b01__Temp_bool_Variable_6__pf,        b01__K2Node_Select_Default_2__pf,        2,
TSwitchPair<bool   ,   ESlateVisibility   >(false,   b01__Temp_byte_Variable_4__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_3__pf)));
                        }
                        if(::IsValid(bpv__TransRoghText__pf))
                        {
                              bpv__TransRoghText__pf-
>SetVisibility(TSwitchValue<bool                ,                ESlateVisibility
>(b01__Temp_bool_Variable_6__pf,        b01__K2Node_Select_Default_2__pf,        2,
TSwitchPair<bool   ,   ESlateVisibility   >(false,   b01__Temp_byte_Variable_4__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_3__pf)));
                        }
                        __CurrentState = -1;
                        break;
                  }
            case 16:
                  {
                        bpfv__CallFunc_Not_PreBool_ReturnValue_1__pf                =
UKismetMathLibrary::Not_PreBool(b01__Temp_bool_Variable_11__pf);
                        b01__Temp_bool_Variable_11__pf                                  =
bpfv__CallFunc_Not_PreBool_ReturnValue_1__pf;
                        __CurrentState = 12;
                        break;
                  }
            case 32:
```

```
                {
                        __CurrentState = 16;
                        break;
                }
        default:
                break;
        }
    } while( __CurrentState != -1 );
    }
    void     UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_7(int32
bpp__EntryPoint__pf)
    {
    bool bpfv__CallFunc_Not_PreBool_ReturnValue_4__pf{};
    int32 __CurrentState = bpp__EntryPoint__pf;
    do
    {
        switch( __CurrentState )
        {
        case 27:
                {
                        if (!b0l__Temp_bool_Variable_15__pf)
                        {
                                __CurrentState = 28;
                                break;
                        }
                }
        case 28:
                {
                        if(::IsValid(bpv__Volume__pf))
                        {
                                bpv__Volume__pf->Settings.RayTracingAO              =
b0l__Temp_bool_Variable_15__pf;
                        }
                        if(::IsValid(bpv__Volume__pf))
                        {
                                bpv__Volume__pf->Settings.bOverride_RayTracingAO    =
true;
                        }
                }
        case 29:
                {
                        b0l__Temp_text_Variable_8__pf                               =
FTextStringHelper::CreateFromBuffer(
    TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"0E9659964BE710F09F66EBA79DA5E484\", \"Ray tracing\")")    );
                        b0l__Temp_text_Variable_9__pf                               =
FTextStringHelper::CreateFromBuffer(
    TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"0981734741926BE189691B9234A4BEC9\", \"Depth\")")    );
                        b0l__Temp_bool_Variable_5__pf                               =
b0l__Temp_bool_Variable_15__pf;
                        if(::IsValid(bpv__Amb__pf))
                        {
                                bpv__Amb__pf->SetText(TSwitchValue<bool    ,    FText
>(b0l__Temp_bool_Variable_5__pf,        b0l__K2Node_Select_Default_8__pf,        2,
TSwitchPair<bool , FText >(false, b0l__Temp_text_Variable_9__pf), TSwitchPair<bool
, FText >(true, b0l__Temp_text_Variable_8__pf)));
                        }
                }
        case 30:
                {
                        b0l__Temp_byte_Variable_9__pf = ESlateVisibility::Visible;
                        b0l__Temp_byte_Variable_10__pf = ESlateVisibility::Hidden;
```

```
                        b01__Temp_bool_Variable_9__pf                                   =
b01__Temp_bool_Variable_15__pf;
                        if(::IsValid(bpv__AOSamles__pf))
                        {
                              bpv__AOSamles__pf->SetVisibility(TSwitchValue<bool  ,
ESlateVisibility >(b01__Temp_bool_Variable_9__pf, b01__K2Node_Select_Default_9__pf,
2, TSwitchPair<bool  , ESlateVisibility >(false, b01__Temp_byte_Variable_10__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_9__pf)));
                        }
                        if(::IsValid(bpv__AOText__pf))
                        {
                              bpv__AOText__pf->SetVisibility(TSwitchValue<bool    ,
ESlateVisibility >(b01__Temp_bool_Variable_9__pf, b01__K2Node_Select_Default_9__pf,
2, TSwitchPair<bool  , ESlateVisibility >(false, b01__Temp_byte_Variable_10__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_9__pf)));
                        }
                        __CurrentState = -1;
                        break;
                  }
            case 31:
                  {
                        bpfv__CallFunc_Not_PreBool_ReturnValue_4__pf              =
UKismetMathLibrary::Not_PreBool(b01__Temp_bool_Variable_15__pf);
                        b01__Temp_bool_Variable_15__pf                            =
bpfv__CallFunc_Not_PreBool_ReturnValue_4__pf;
                        __CurrentState = 27;
                        break;
                  }
            case 35:
                  {
                        __CurrentState = 31;
                        break;
                  }
            default:
                  break;
            }
      } while( __CurrentState != -1 );
      }
      void      UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_8(int32
bpp__EntryPoint__pf)
      {
      bool bpfv__CallFunc_Not_PreBool_ReturnValue_3__pf{};
      int32 __CurrentState = bpp__EntryPoint__pf;
      do
      {
            switch( __CurrentState )
            {
            case 22:
                  {
                        if (!b01__Temp_bool_Variable_14__pf)
                        {
                              __CurrentState = 23;
                              break;
                        }
                  }
            case 23:
                  {
                        b01__Temp_byte_Variable_1__pf                             =
ERayTracingGlobalIlluminationType::FinalGather;
                        b01__Temp_byte_Variable_2__pf                             =
ERayTracingGlobalIlluminationType::Disabled;
                        b01__Temp_bool_Variable__pf                               =
b01__Temp_bool_Variable_14__pf;
                        if(::IsValid(bpv__Volume__pf))
```

```
                        {
                              bpv__Volume__pf->Settings.RayTracingGIType          =
TSwitchValue<bool                 ,                ERayTracingGlobalIlluminationType
>(b0l__Temp_bool_Variable__pf,         b0l__K2Node_Select_Default_5__pf,         2,
TSwitchPair<bool         ,         ERayTracingGlobalIlluminationType         >(false,
b0l__Temp_byte_Variable_2__pf),                  TSwitchPair<bool                  ,
ERayTracingGlobalIlluminationType >(true, b0l__Temp_byte_Variable_1__pf));
                        }
                        if(::IsValid(bpv__Volume__pf))
                        {
                              bpv__Volume__pf->Settings.bOverride_RayTracingGI    =
true;
                        }
                  }
            case 24:
                  {
                        b0l__Temp_text_Variable_6__pf                               =
FTextStringHelper::CreateFromBuffer(
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"0E9659964BE710F09F66EBA79DA5E484\", \"Ray tracing\")")    );
                        b0l__Temp_text_Variable_7__pf                               =
FTextStringHelper::CreateFromBuffer(
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"F474195143F3C00C6500D7AE313BBE3C\", \"NO GI\")")    );
                        b0l__Temp_bool_Variable_4__pf                               =
b0l__Temp_bool_Variable_14__pf;
                        if(::IsValid(bpv__Glob__pf))
                        {
                              bpv__Glob__pf->SetText(TSwitchValue<bool    ,    FText
>(b0l__Temp_bool_Variable_4__pf,         b0l__K2Node_Select_Default_6__pf,         2,
TSwitchPair<bool , FText >(false, b0l__Temp_text_Variable_7__pf), TSwitchPair<bool
, FText >(true, b0l__Temp_text_Variable_6__pf)));
                        }
                  }
            case 25:
                  {
                        b0l__Temp_byte_Variable_7__pf = ESlateVisibility::Visible;
                        b0l__Temp_byte_Variable_8__pf = ESlateVisibility::Hidden;
                        b0l__Temp_bool_Variable_8__pf                               =
b0l__Temp_bool_Variable_14__pf;
                        if(::IsValid(bpv__GIraystext__pf))
                        {
                              bpv__GIraystext__pf->SetVisibility(TSwitchValue<bool
,                ESlateVisibility                >(b0l__Temp_bool_Variable_8__pf,
b0l__K2Node_Select_Default_7__pf, 2, TSwitchPair<bool , ESlateVisibility >(false,
b0l__Temp_byte_Variable_8__pf),   TSwitchPair<bool   ,   ESlateVisibility   >(true,
b0l__Temp_byte_Variable_7__pf)));
                        }
                        if(::IsValid(bpv__GIRays__pf))
                        {
                              bpv__GIRays__pf->SetVisibility(TSwitchValue<bool    ,
ESlateVisibility >(b0l__Temp_bool_Variable_8__pf, b0l__K2Node_Select_Default_7__pf,
2,  TSwitchPair<bool  ,  ESlateVisibility  >(false,  b0l__Temp_byte_Variable_8__pf),
TSwitchPair<bool , ESlateVisibility >(true, b0l__Temp_byte_Variable_7__pf)));
                        }
                        __CurrentState = -1;
                        break;
                  }
            case 26:
                  {
                        bpfv__CallFunc_Not_PreBool_ReturnValue_3__pf                =
UKismetMathLibrary::Not_PreBool(b0l__Temp_bool_Variable_14__pf);
                        b0l__Temp_bool_Variable_14__pf                              =
bpfv__CallFunc_Not_PreBool_ReturnValue_3__pf;
```

```
                        __CurrentState = 22;
                        break;
                  }
            case 34:
                  {
                        __CurrentState = 26;
                        break;
                  }
            default:
                  break;
            }
      } while( __CurrentState != -1 );
      }
      void      UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_9(int32
bpp__EntryPoint__pf)
      {
      bool bpfv__CallFunc_Not_PreBool_ReturnValue_2__pf{};
      int32 __CurrentState = bpp__EntryPoint__pf;
      do
      {
            switch( __CurrentState )
            {
            case 17:
                  {
                        bpfv__CallFunc_Not_PreBool_ReturnValue_2__pf              =
UKismetMathLibrary::Not_PreBool(b0l__Temp_bool_Variable_12__pf);
                        b0l__Temp_bool_Variable_12__pf                            =
bpfv__CallFunc_Not_PreBool_ReturnValue_2__pf;
                  }
            case 18:
                  {
                        if (!b0l__Temp_bool_Variable_12__pf)
                        {
                              __CurrentState = 19;
                              break;
                        }
                  }
            case 19:
                  {
                        b0l__Temp_byte_Variable__pf                               =
EReflectionsType::ScreenSpace;
                        b0l__Temp_byte_Variable_11__pf                            =
EReflectionsType::RayTracing;
                        b0l__Temp_bool_Variable_16__pf                            =
b0l__Temp_bool_Variable_12__pf;
                        if(::IsValid(bpv__Volume__pf))
                        {
                              bpv__Volume__pf->Settings.ReflectionsType           =
TSwitchValue<bool     ,     EReflectionsType     >(b0l__Temp_bool_Variable_16__pf,
b0l__K2Node_Select_Default_10__pf, 2, TSwitchPair<bool , EReflectionsType >(false,
b0l__Temp_byte_Variable__pf),   TSwitchPair<bool    ,   EReflectionsType   >(true,
b0l__Temp_byte_Variable_11__pf));
                        }
                        if(::IsValid(bpv__Volume__pf))
                        {
                              bpv__Volume__pf->Settings.bOverride_ReflectionsType
= true;
                        }
                  }
            case 20:
                  {
                        b0l__Temp_text_Variable_4__pf                             =
FTextStringHelper::CreateFromBuffer(
```

```
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"0E9659964BE710F09F66EBA79DA5E484\", \"Ray tracing\")")    );
                        b01__Temp_text_Variable_5__pf                                =
FTextStringHelper::CreateFromBuffer(
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"4E2835164A1553889672198C5D1ACBBE\", \"SSR\")")      );
                        b01__Temp_bool_Variable_3__pf                                =
b01__Temp_bool_Variable_12__pf;
                        if(::IsValid(bpv__Refle__pf))
                        {
                              bpv__Refle__pf->SetText(TSwitchValue<bool   ,   FText
>(b01__Temp_bool_Variable_3__pf,        b01__K2Node_Select_Default_3__pf,        2,
TSwitchPair<bool , FText >(false, b01__Temp_text_Variable_5__pf), TSwitchPair<bool
, FText >(true, b01__Temp_text_Variable_4__pf)));
                        }
                  }
            case 21:
                  {
                        b01__Temp_byte_Variable_5__pf = ESlateVisibility::Visible;
                        b01__Temp_byte_Variable_6__pf = ESlateVisibility::Hidden;
                        b01__Temp_bool_Variable_7__pf                                =
b01__Temp_bool_Variable_12__pf;
                        if(::IsValid(bpv__RefRo__pf))
                        {
                              bpv__RefRo__pf->SetVisibility(TSwitchValue<bool     ,
ESlateVisibility >(b01__Temp_bool_Variable_7__pf, b01__K2Node_Select_Default_4__pf,
2,  TSwitchPair<bool  ,  ESlateVisibility  >(false,  b01__Temp_byte_Variable_6__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_5__pf)));
                        }
                        if(::IsValid(bpv__RefRaysText__pf))
                        {
                              bpv__RefRaysText__pf-
>SetVisibility(TSwitchValue<bool                    ,                    ESlateVisibility
>(b01__Temp_bool_Variable_7__pf,        b01__K2Node_Select_Default_4__pf,        2,
TSwitchPair<bool   ,   ESlateVisibility   >(false,   b01__Temp_byte_Variable_6__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_5__pf)));
                        }
                        if(::IsValid(bpv__REROGHTExt__pf))
                        {
                              bpv__REROGHTExt__pf->SetVisibility(TSwitchValue<bool
,                ESlateVisibility                >(b01__Temp_bool_Variable_7__pf,
b01__K2Node_Select_Default_4__pf, 2, TSwitchPair<bool , ESlateVisibility >(false,
b01__Temp_byte_Variable_6__pf),   TSwitchPair<bool   ,   ESlateVisibility   >(true,
b01__Temp_byte_Variable_5__pf)));
                        }
                        if(::IsValid(bpv__RefRays__pf))
                        {
                              bpv__RefRays__pf->SetVisibility(TSwitchValue<bool   ,
ESlateVisibility >(b01__Temp_bool_Variable_7__pf, b01__K2Node_Select_Default_4__pf,
2,  TSwitchPair<bool  ,  ESlateVisibility  >(false,  b01__Temp_byte_Variable_6__pf),
TSwitchPair<bool , ESlateVisibility >(true, b01__Temp_byte_Variable_5__pf)));
                        }
                        __CurrentState = -1;
                        break;
                  }
            case 33:
                  {
                        __CurrentState = 17;
                        break;
                  }
            default:
                  break;
            }
      } while( __CurrentState != -1 );
```

```
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_10(int32
bpp__EntryPoint__pf)
	{
	APlayerController* bpfv__CallFunc_GetPlayerController_ReturnValue_2__pf{};
	check(bpp__EntryPoint__pf == 41);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_GetPlayerController_ReturnValue_2__pf                          =
UGameplayStatics::GetPlayerController(this, 0);
	if(::IsValid(bpfv__CallFunc_GetPlayerController_ReturnValue_2__pf))
	{
		bpfv__CallFunc_GetPlayerController_ReturnValue_2__pf-
>AController::Possess(bpv__Pawn3__pf);
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_11(int32
bpp__EntryPoint__pf)
	{
	APlayerController* bpfv__CallFunc_GetPlayerController_ReturnValue_1__pf{};
	check(bpp__EntryPoint__pf == 39);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_GetPlayerController_ReturnValue_1__pf                          =
UGameplayStatics::GetPlayerController(this, 0);
	if(::IsValid(bpfv__CallFunc_GetPlayerController_ReturnValue_1__pf))
	{
		bpfv__CallFunc_GetPlayerController_ReturnValue_1__pf-
>AController::Possess(bpv__Pawn2__pf);
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_12(int32
bpp__EntryPoint__pf)
	{
	APlayerController* bpfv__CallFunc_GetPlayerController_ReturnValue__pf{};
	check(bpp__EntryPoint__pf == 37);
	// optimized KCST_UnconditionalGoto
	bpfv__CallFunc_GetPlayerController_ReturnValue__pf                            =
UGameplayStatics::GetPlayerController(this, 0);
	if(::IsValid(bpfv__CallFunc_GetPlayerController_ReturnValue__pf))
	{
		bpfv__CallFunc_GetPlayerController_ReturnValue__pf-
>AController::Possess(bpv__Pawn1__pf);
	}
	return; //KCST_EndOfThread
	}
	void	UShadows_C__pf2937682852::bpf__ExecuteUbergraph_Shadows__pf_13(int32
bpp__EntryPoint__pf)
	{
	bool bpfv__CallFunc_Not_PreBool_ReturnValue__pf{};
	int32 bpfv__CallFunc_Add_IntInt_ReturnValue__pf{};
	int32 bpfv__CallFunc_Array_Length_ReturnValue__pf{};
	bool bpfv__CallFunc_Less_IntInt_ReturnValue__pf{};
	TArray< int32, TInlineAllocator<8> > __StateStack;

	int32 __CurrentState = bpp__EntryPoint__pf;
	do
	{
		switch( __CurrentState )
		{
		case 1:
			{
				bpfv__CallFunc_Not_PreBool_ReturnValue__pf                =
UKismetMathLibrary::Not_PreBool(b0l__Temp_bool_Variable_10__pf);
```

```
                        b01__Temp_bool_Variable_10__pf                                  =
bpfv__CallFunc_Not_PreBool_ReturnValue__pf;
                  }
            case 2:
                  {
                        if (!b01__Temp_bool_Variable_10__pf)
                        {
                              __CurrentState = 3;
                              break;
                        }
                  }
            case 3:
                  {
                        (b01__CallFunc_GetAllActorsOfClass_OutActors__pf).Reset();
                        UGameplayStatics::GetAllActorsOfClass(this,
ALight::StaticClass(),                                                          /*out*/
TArrayCaster<ALight*>(b01__CallFunc_GetAllActorsOfClass_OutActors__pf).Get<AActor*
>());
                  }
            case 4:
                  {
                        b01__Temp_int_Loop_Counter_Variable__pf = 0;
                  }
            case 5:
                  {
                        b01__Temp_int_Array_Index_Variable__pf = 0;
                  }
            case 6:
                  {
                        bpfv__CallFunc_Array_Length_ReturnValue__pf                =
FCustomThunkTemplates::Array_Length(b01__CallFunc_GetAllActorsOfClass_OutActors__p
f);
                        bpfv__CallFunc_Less_IntInt_ReturnValue__pf                 =
UKismetMathLibrary::Less_IntInt(b01__Temp_int_Loop_Counter_Variable__pf,
bpfv__CallFunc_Array_Length_ReturnValue__pf);
                        if (!bpfv__CallFunc_Less_IntInt_ReturnValue__pf)
                        {
                              __CurrentState   =   (__StateStack.Num()   >   0)   ?
__StateStack.Pop(/*bAllowShrinking=*/ false) : -1;
                              break;
                        }
                  }
            case 7:
                  {
                        b01__Temp_int_Array_Index_Variable__pf                     =
b01__Temp_int_Loop_Counter_Variable__pf;
                  }
            case 8:
                  {
                        __StateStack.Push(11);
                  }
            case 9:
                  {

      FCustomThunkTemplates::Array_Get(b01__CallFunc_GetAllActorsOfClass_OutActors
__pf,                b01__Temp_int_Array_Index_Variable__pf,                /*out*/
b01__CallFunc_Array_Get_Item__pf);
                        if(::IsValid(b01__CallFunc_Array_Get_Item__pf)           &&
::IsValid((*(AccessPrivateProperty<ULightComponent*
>((b01__CallFunc_Array_Get_Item__pf), ALight::__PPO__LightComponent() )))))
                        {
                              (*(AccessPrivateProperty<ULightComponent*
>((b01__CallFunc_Array_Get_Item__pf),     ALight::__PPO__LightComponent()     )))-
>ULightComponentBase::SetCastRaytracedShadow(b01__Temp_bool_Variable_10__pf);
```

```
                        }
                  }
            case 10:
                  {
                        b0l__Temp_text_Variable__pf                                    =
FTextStringHelper::CreateFromBuffer(
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"0E9659964BE710F09F66EBA79DA5E484\", \"Ray tracing\")")    );
                        b0l__Temp_text_Variable_1__pf                                  =
FTextStringHelper::CreateFromBuffer(
      TEXT("NSLOCTEXT(\"[CE5421364A4A8B60B1D4068D325DE581]\",
\"4C63B2E7440A85BB884217BCA157DBE2\", \"Raster\")")   );
                        b0l__Temp_bool_Variable_1__pf                                  =
b0l__Temp_bool_Variable_10__pf;
                        if(::IsValid(bpv__DR__pf))
                        {
                              bpv__DR__pf->SetText(TSwitchValue<bool     ,     FText
>(b0l__Temp_bool_Variable_1__pf,         b0l__K2Node_Select_Default__pf,         2,
TSwitchPair<bool , FText >(false, b0l__Temp_text_Variable_1__pf), TSwitchPair<bool
, FText >(true, b0l__Temp_text_Variable__pf)));
                        }
                        __CurrentState    =    (__StateStack.Num()    >    0)    ?
__StateStack.Pop(/*bAllowShrinking=*/ false) : -1;
                        break;
                  }
            case 11:
                  {
                        bpfv__CallFunc_Add_IntInt_ReturnValue__pf                      =
UKismetMathLibrary::Add_IntInt(b0l__Temp_int_Loop_Counter_Variable__pf, 1);
                        b0l__Temp_int_Loop_Counter_Variable__pf                        =
bpfv__CallFunc_Add_IntInt_ReturnValue__pf;
                        __CurrentState = 6;
                        break;
                  }
            case 36:
                  {
                        __CurrentState = 1;
                        break;
                  }
            default:
                  check(false); // Invalid state
                  break;
            }
      } while( __CurrentState != -1 );
      }
```

ДОДАТОК Б
(довідковий)

«АПРОБАЦІЯ РЕЗУЛЬТАТІВ РОБОТИ»

МАТЕРІАЛИ МІЖНАРОДНОЇ НАУКОВОЇ КОНФЕРЕЦІЇ

СТРАТЕГІЧНІ НАПРЯМИ РОЗВИТКУ НАУКИ: ФАКТОРИ ВПЛИВУ ТА ВЗАЄМОДІЇ

ТЕХНІЧНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Суми (Україна)

22.05.2020

МЕТОДЫ И ПРОБЛЕМЫ ВИЗУАЛИЗАЦИИ 3D ГЕОМЕТРИИ В РЕАЛЬНОМ ВРЕМЕНИ

Засименко Георгий Романович

Здобувач вищої освіти факультету комп'ютерних наук

Харківський національний університет радіоелектроніки

Україна

В подавляющем большинстве систем визуализации в реальном времени используется алгоритм отложенного затенения [1] (рис. 1).

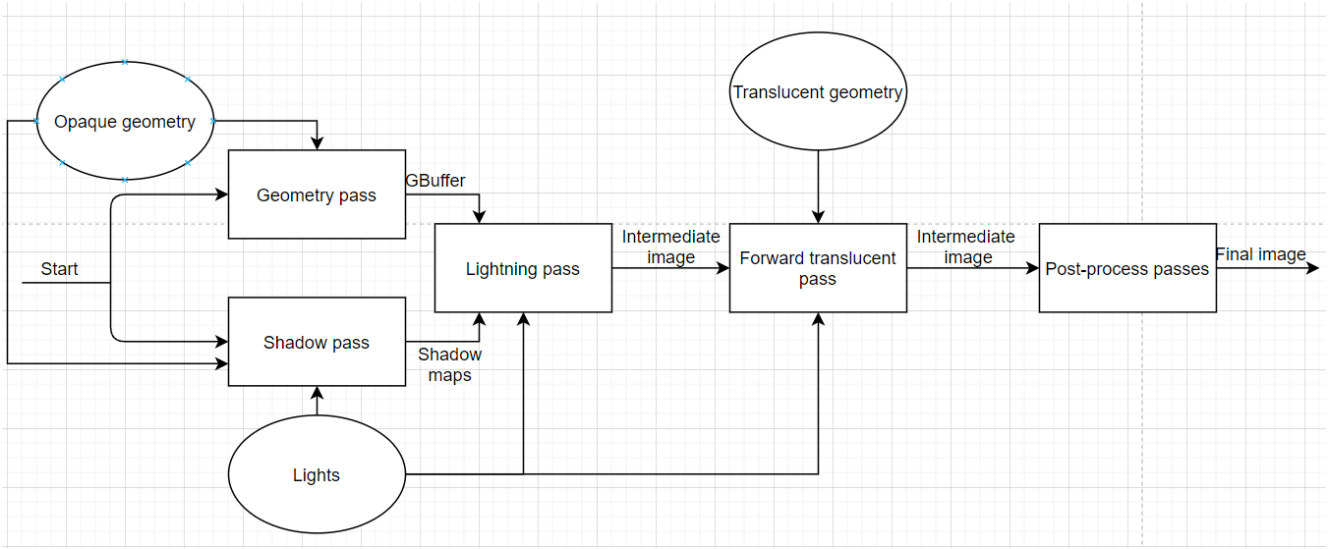

Рис. 1. **Алгоритм отложенного затенения**

Первым этапом является генерация карт теней (shadow pass). Тени – это результат отсутствия света в связи с тем, что луч света не достигает поверхности (рис. 2). Для достижение этого эффекта из положения источника освещения рассчитывается глубина геометрии сцены [1].

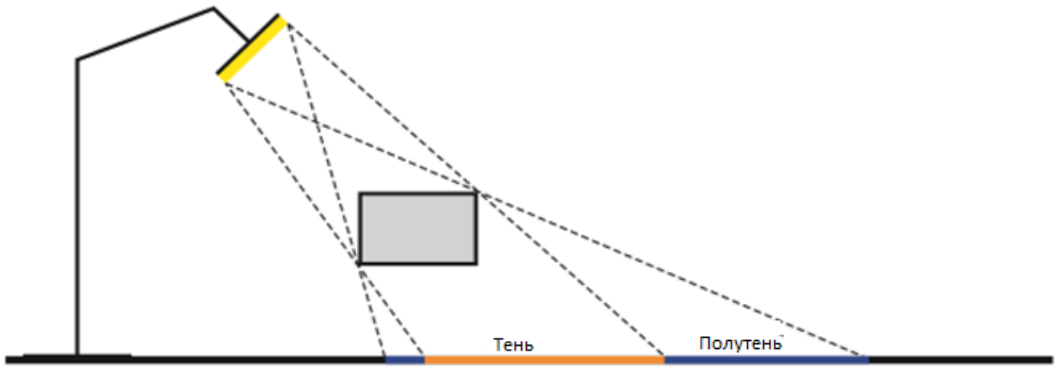

Рис. 2. **Тени**

Это достигается путем постановки виртуальной камеры в положение источника освещения и растеризации геометрии сцены в буфер глубины (рис. 3), чем темнее пиксель – тем ближе геометрия к камере.

Рис. 3. **Буфер глубины**

Получение карт глубины метод растеризации хорошо работает для бесконечно маленьких аналитических источников освещения, но в реальном мире не существует таковых. Тень от источников, у которых есть площадь имеет так же полутень. Чтобы получить полутень от, например, прямоугольного источника освещения можно получить карты теней с каждого угла и потом обрабатывать пиксель в зависимости от количества карт теней, в которые он попал. Если попал во все – находится в тени, если только в несколько – в полутени (рис. 2). Для источников неправильной формы такой способ не подойдет.

С появлением аппаратной трассировки лучей тени в реальном времени можно получить этим способом. Одним из вариантов является так же постановка виртуальной камеры в положение источника освещения и запись в глубины в буфер. Получение теней от источников, которые имеют площадь таким способом легче, потому что можно сделать обратную трассировку луча в случайную точку площади источника освещения.

На этапе геометрии (geometry pass) происходит генерация геометрического буфера, который является совокупностью буферов параметров геометрии сцены.

Например, для модели освещения PBR [2] этот буфер будет состоять из таких буферов (рис. 4):

– буфер позиции;

– буфер шероховатости;

– альбедо буфер;

– буфер нормалей;

– буфер металличности.

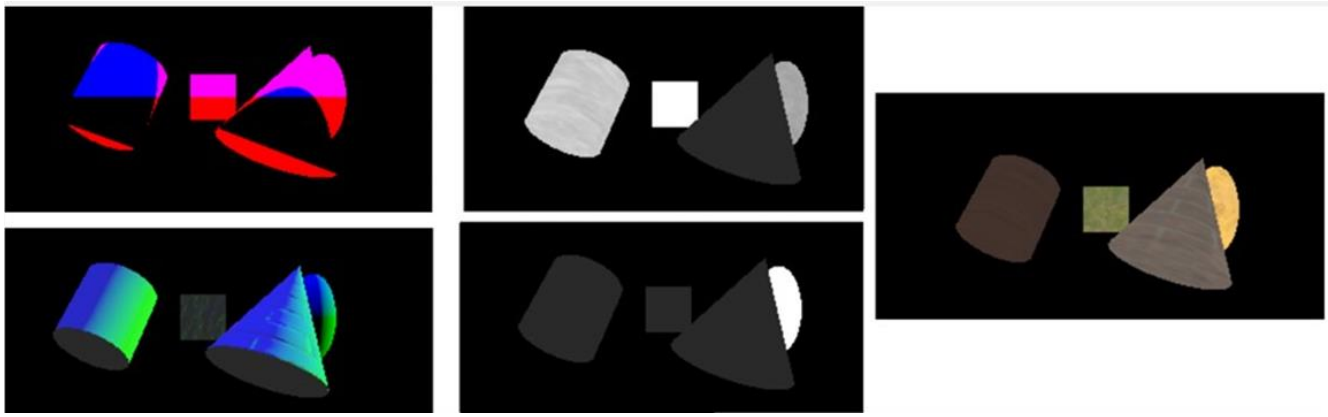

Рис. 4. **Геометрический буфер**

Следующим этапом является освещение (lightning pass). Для расчета освещения каждого пикселя для каждого источника освещения находится положение на геометрическом буфере, где источник взаимодействует с геометрией и имея все параметры геометрии в буфере легко найти цвет пикселя. Например, точечный источник освещения может быть представлен, как сфера и можно найти, какую геометрию он освещает (рис. 5).

Можно сказать, что такой подход – это постобработка изображения освещением. Такой подход имеет алгоритмическую сложность O (n + m), где n – количество геометрии, а m – количество источников освещения.

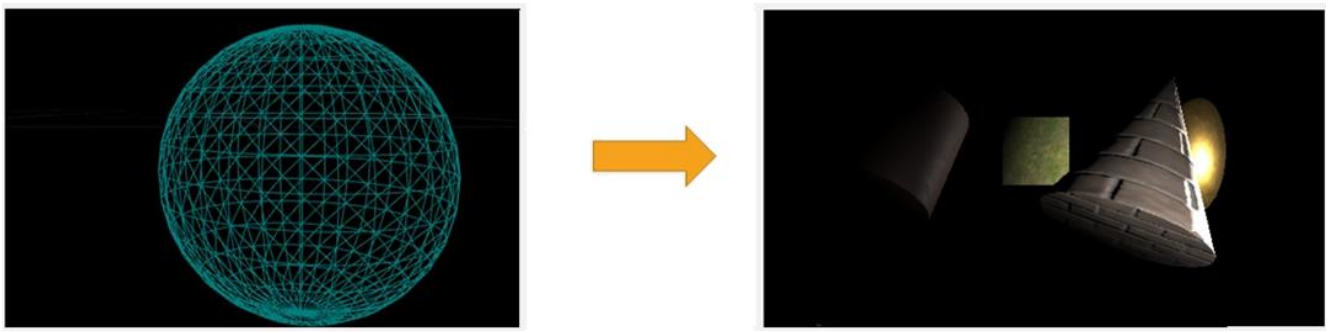

Рис. 5. **Точечный источник освещения**

Следующий этап – прозрачность (forward translucent pass). Прозрачная геометрия накладывается поверх непрозрачной, т.е. информация о непрозрачной геометрии должна быть уже получена. Прозрачная геометрия должна преломлять свет согласно закону преломления света на границе раздела сред. Информация о геометрии куда преломится луч не всегда присутствует в геометрическом буфере, потому что луч может отразиться в геометрию вне поля зрения.

Еще одна проблема возникает, когда необходимо отрисовать, например, прозрачную сферу. Методом растеризации не ясно какую сторону сферы необходимо отображать в первую очередь, не говоря уже о том, что повлиять на это крайне сложно в графических API. Проблему можно решить имея две полусферы, одну для передней части сферы, другую для обратной и всегда поворачивать их «лицом» к камере, но сфера – это идеальный пример, со случайно геометрией это повторить нельзя. В связи с этим возникают артефакты преломления света (рис. 6).

Так же большой проблемой являются отражения. В классическом обратном методе затенения это делается одним из этапов пост процесса. Методом растеризации нет способа получить корректные отражения. В большом количестве систем используется SSLR [1]. В геометрическом буфере есть информация о нормалях геометрии и глубине сцены, поэтому можно аппроксимировать на гладкой поверхности то, что отражается. Из названия алгоритма ясно, что этот эффект работает только на геометрию, которая видна в текущем поле зрения, поэтому с помощью этого алгоритма невозможно сделать что-то похожее на зеркало.

Рис. 6. **Артефакты преломления света методом растеризации**

Алгоритм подходит, когда на сцене много геометрических объектов неправильной формы и отображения размыты, когда такое отражение попадает на равный объект вблизи камеры, то артефакт трудно не заметить (рис. 7).

Из рисунка видно, что на металлическом стакане отражается только та часть книг, которая входит в поле зрения, а на большей части стакана вообще нет отражений, потому что то, что могло бы там отразиться не входит в поле зрения.

На этапах постобработки (post-process passes) выполняется различные эффекты конечного изображения, например размытие, и приведение его к формату, который подходит к выводу, например, приведение изображения к LDR с HDR [1].

Появление аппаратной поддержки трассировки лучшей позволит решить проблемы, связанные с неправильным преломлением луча на границе сред методом растеризации, а также позволит иметь физически корректные отражения на поверхностях, так как в методе трассировки лучшей [3] взаимодействие с поверхностями лежит в сердце алгоритма. Т.е. каждый выпущенный луч имеет возможность к преломлению на границе сред, а также после этого без каких-либо проблем может отразиться от гладкой поверхности.

Рис. 7. **Артефакты метода SSLR**

Проблема этого способа состоит в том, что для него необходимы большие вычислительные мощности. Современные видеокарты начиная с NVIDIA Turing пытаются решить эту проблему.

Выводы. В результате, проблема артефактов имеет варианты удовлетворительных решений, которые в свою очередь нуждаются в дальнейшем тестировании на пригодность работы в реальных системах визуализации в реальном времени.

Список использованных источников:

1. Van Verth J. (2013). Essential mathematic for games and interactive applications, Third edition. Boca Raton: CRC Press.

2. Pharr M. (2017). Physically based rendering from theory to implementation. MA: Elsevier.

3. Suffern K. (2007), Ray tracing from the ground up. Boca Raton: CRC Press.

ДОДАТОК В
(довідковий)

«СЛАЙДИ ПРЕЗЕНТАЦІЇ»

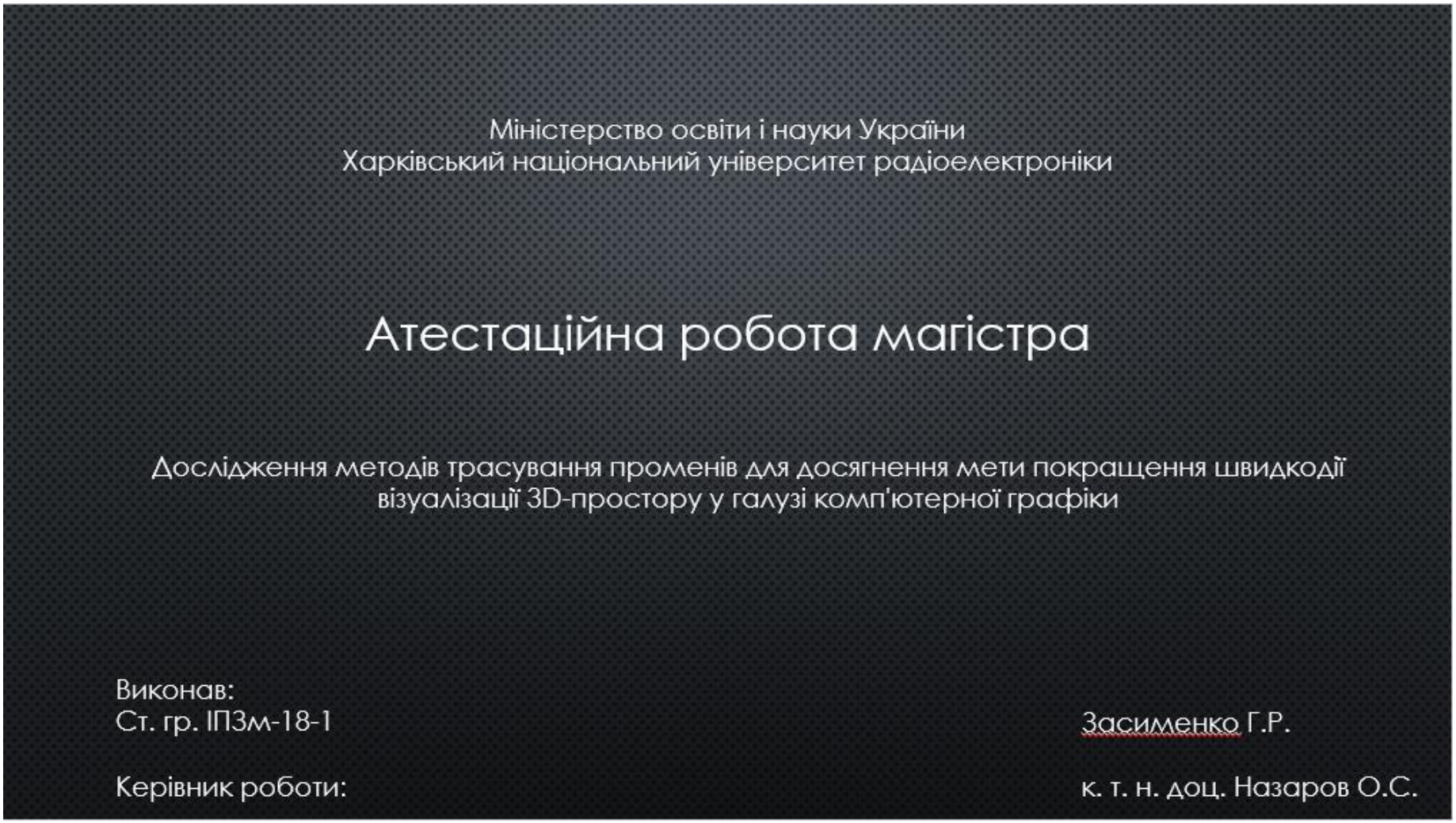

Рисунок А.1 – Титульний слайд

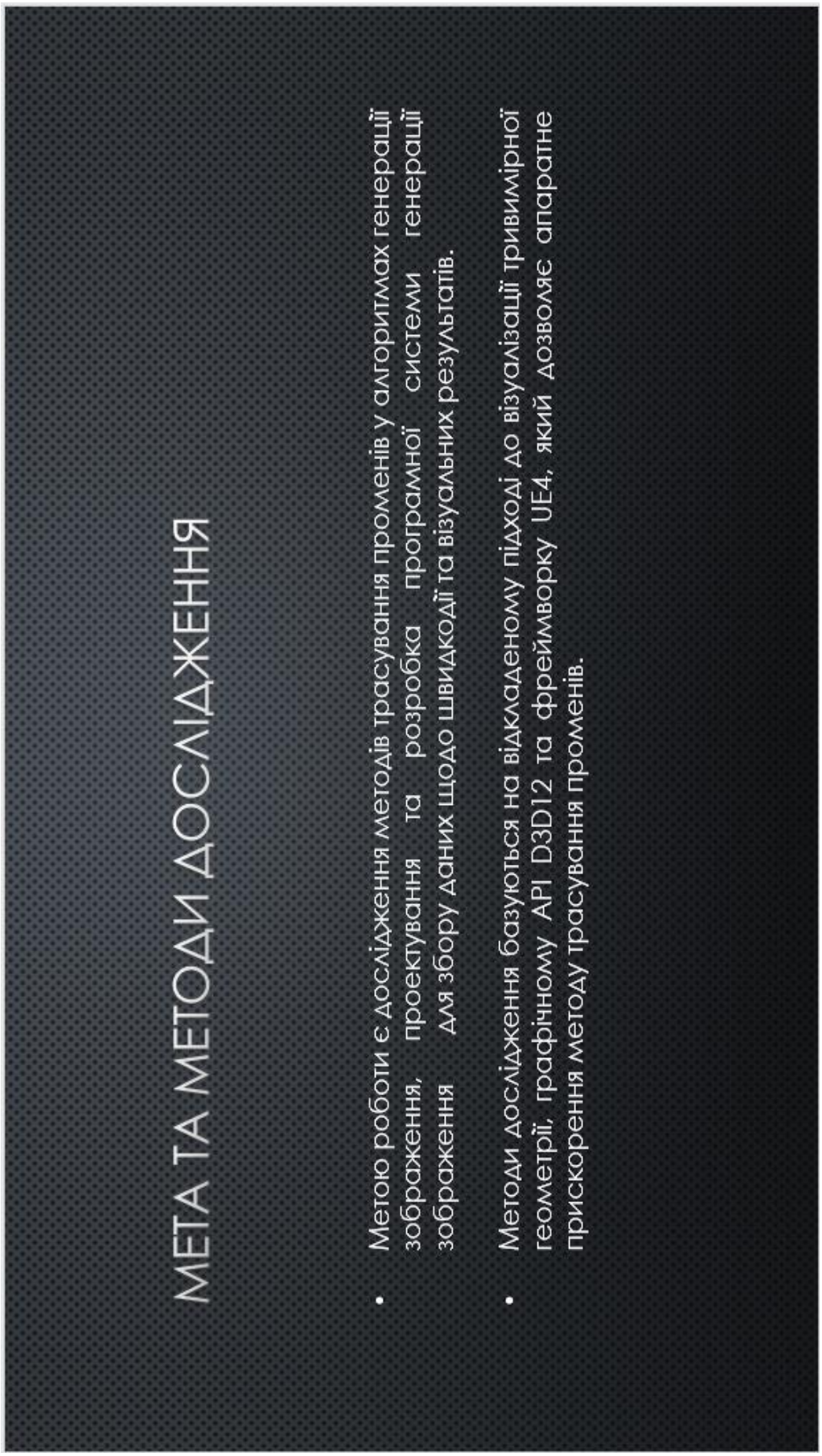

Рисунок А.2 – Мета та методи дослідження

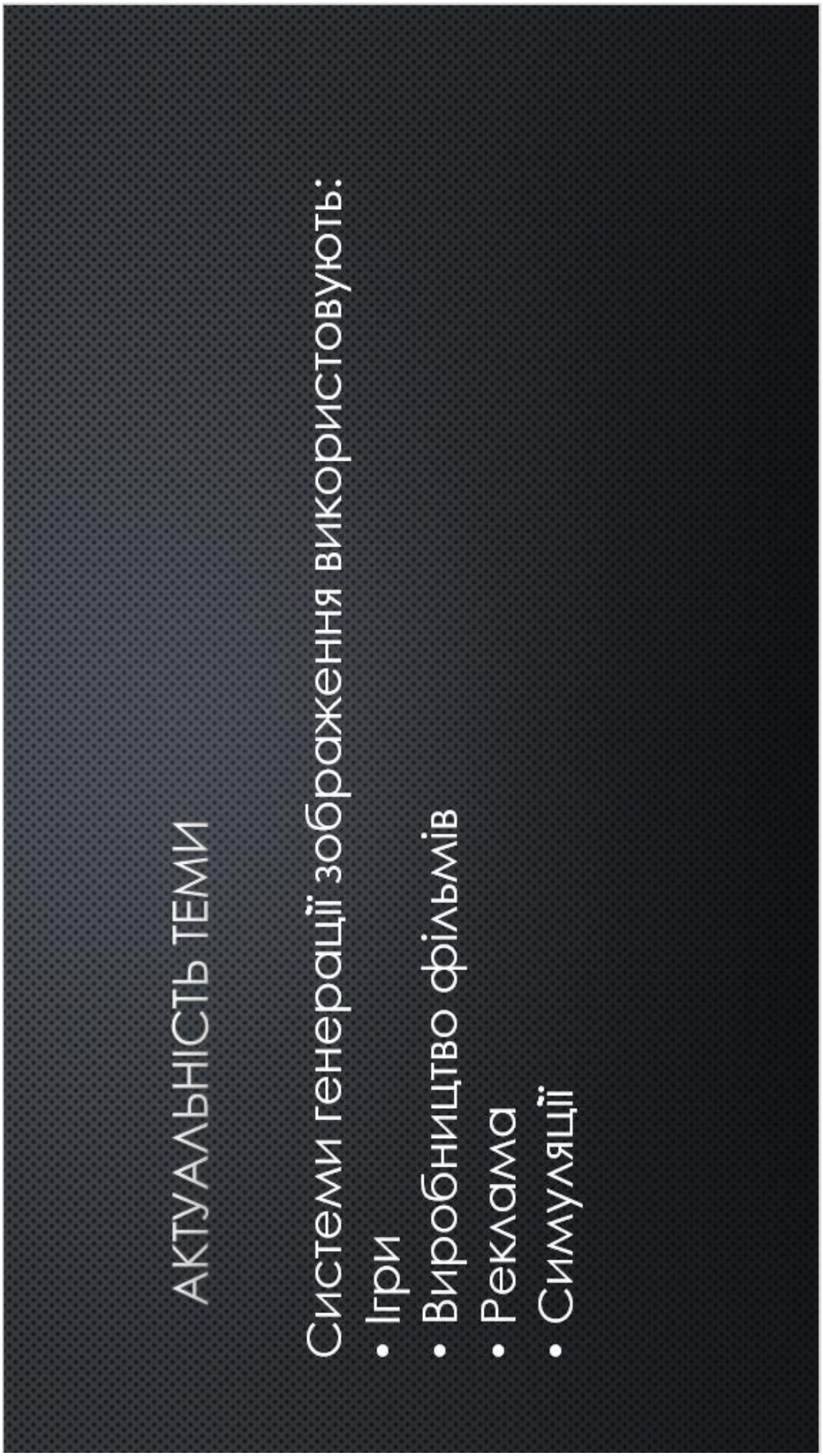

Рисунок А.3 – Актуальність теми

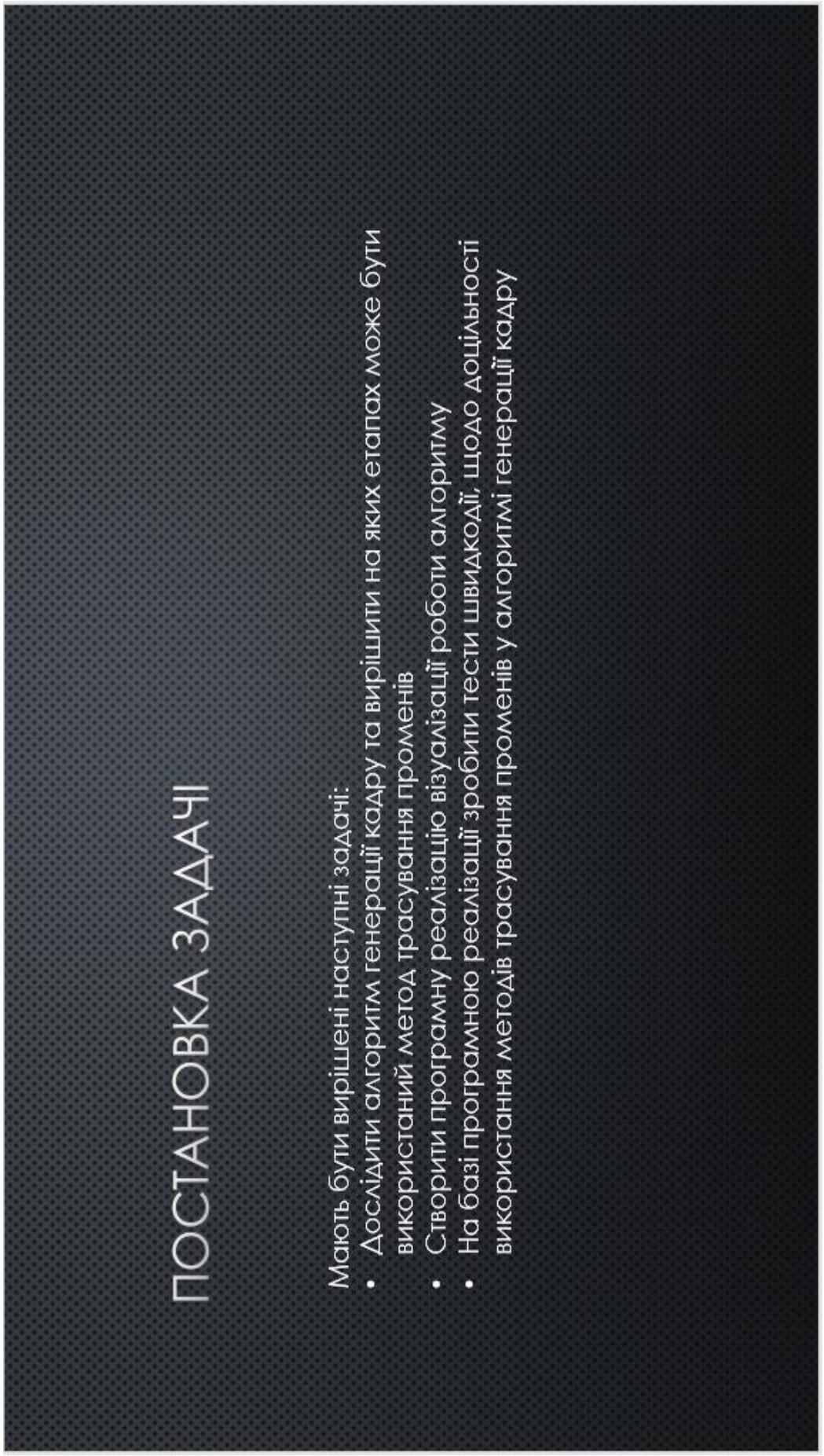

Рисунок А.4 – Постановка задачі

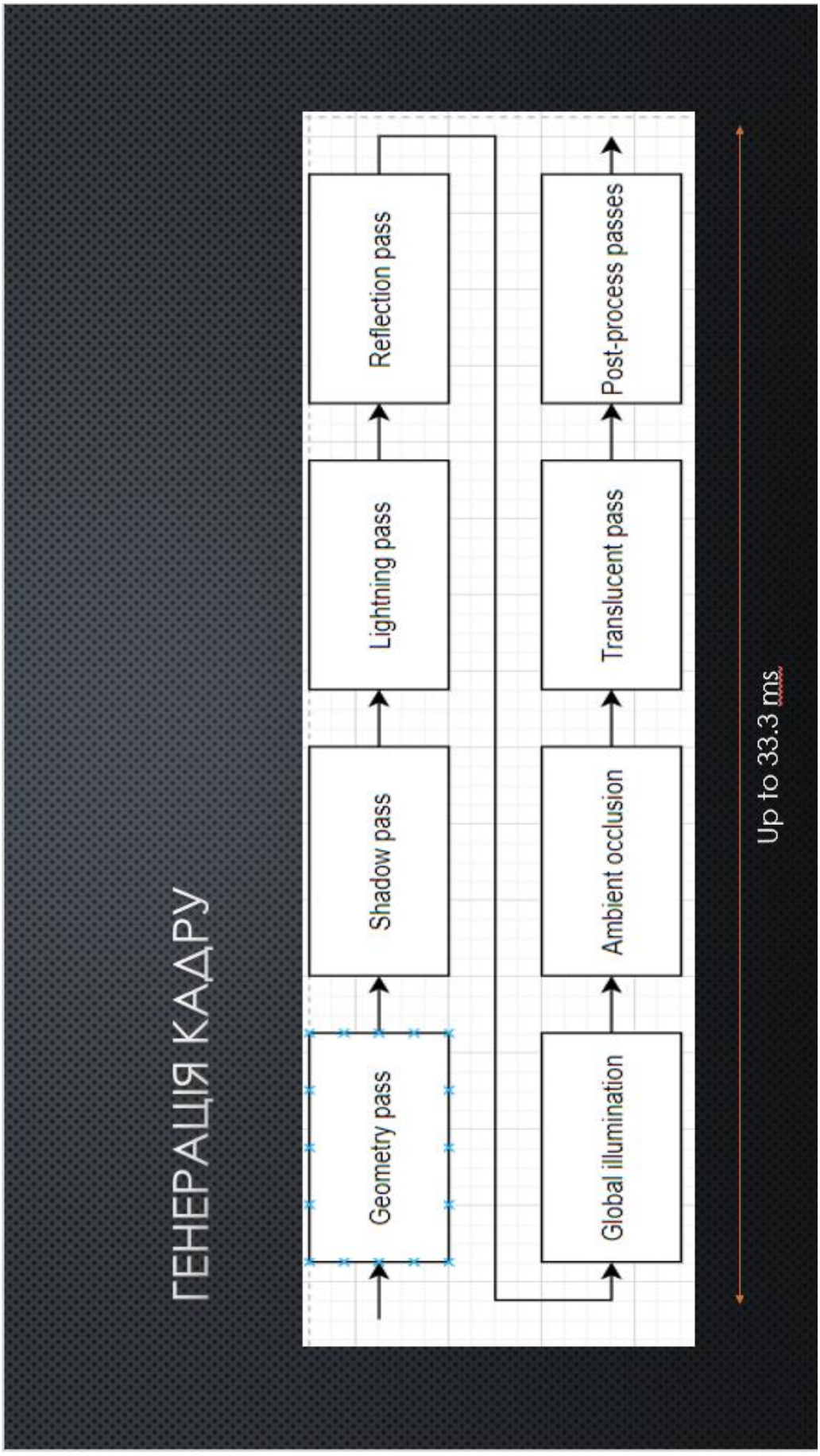

Рисунок А.5 – Генерація кадру

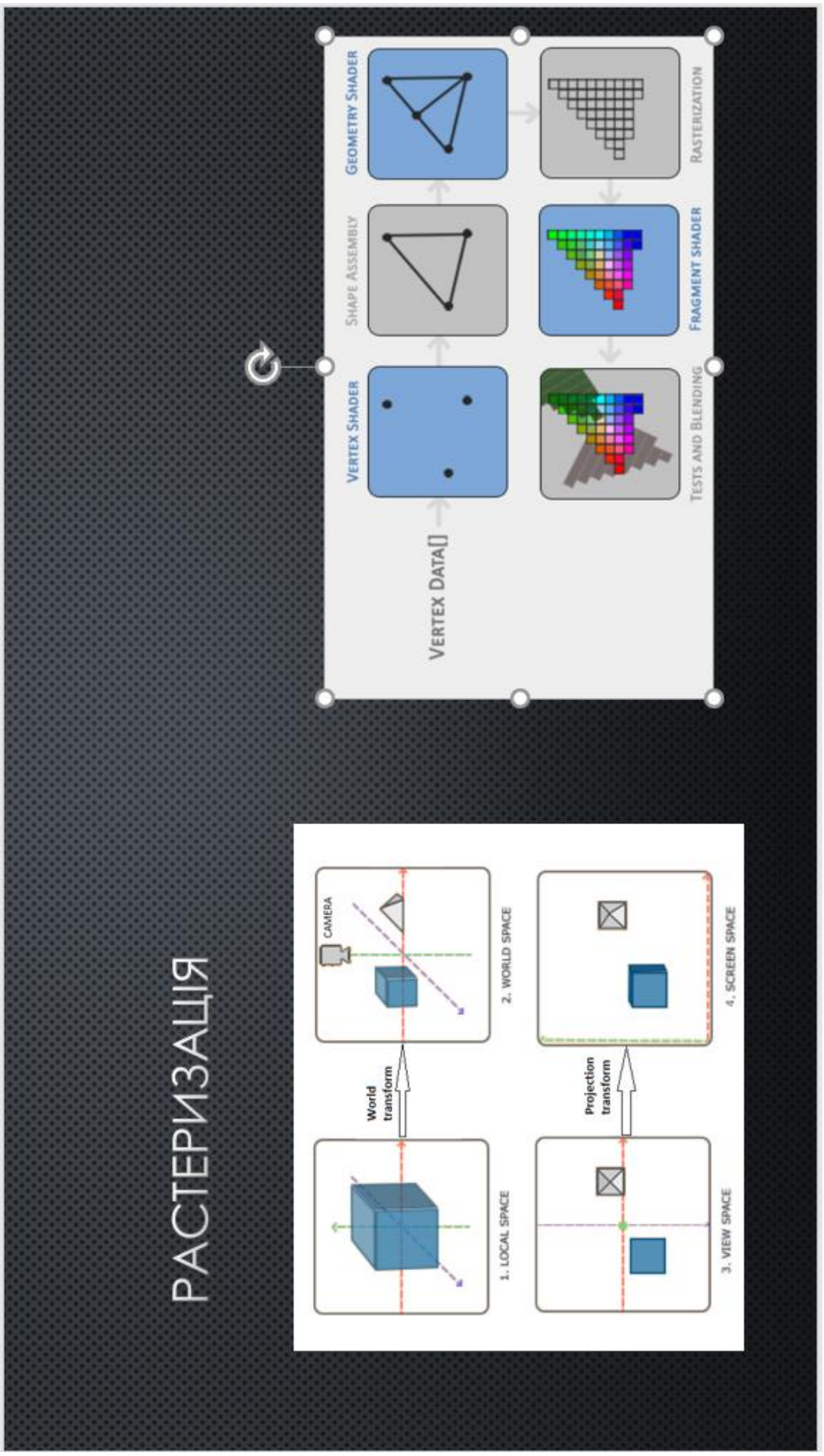

Рисунок А.6 – Растеризація

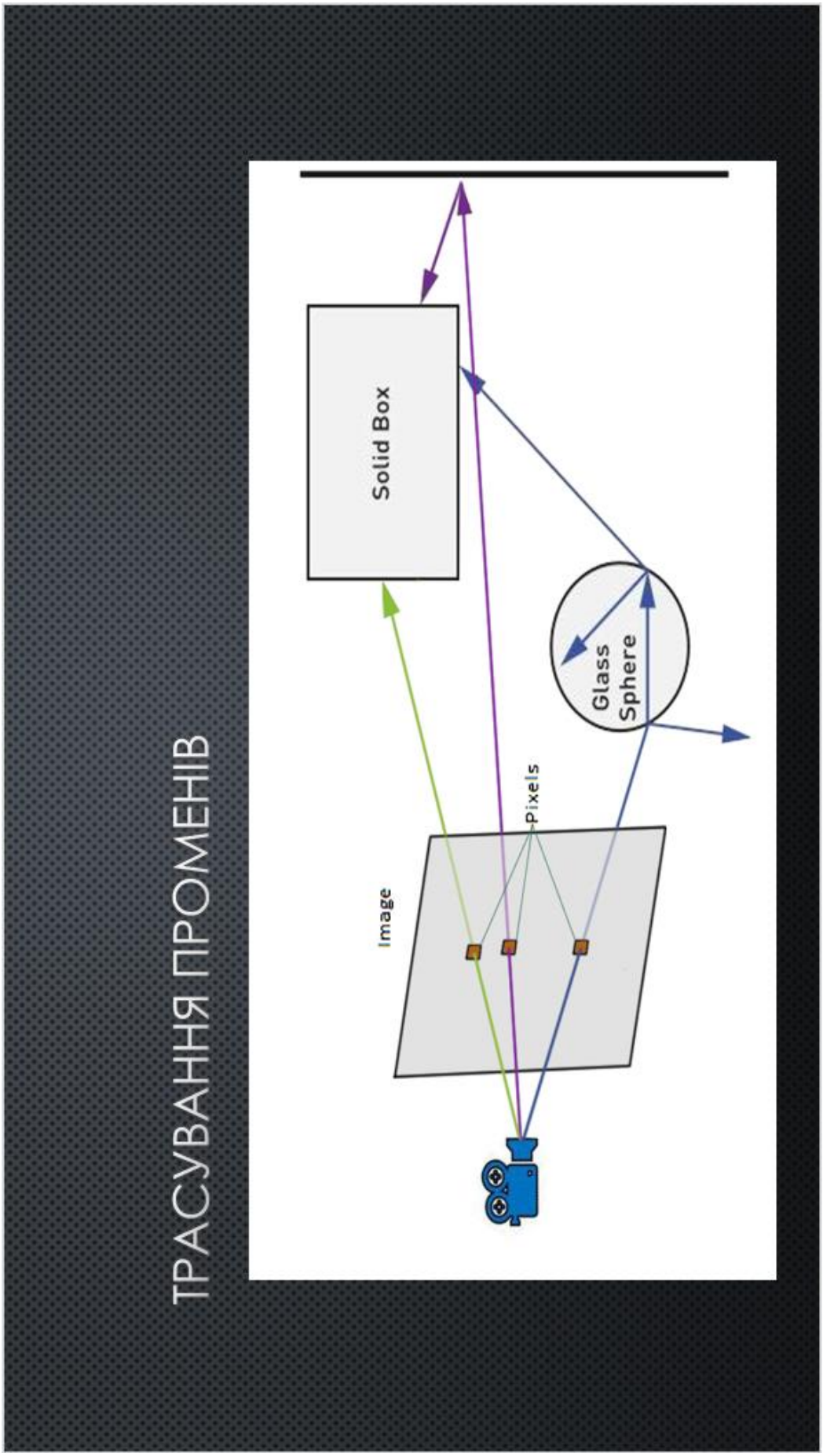

Рисунок А.7 – Трасування променів

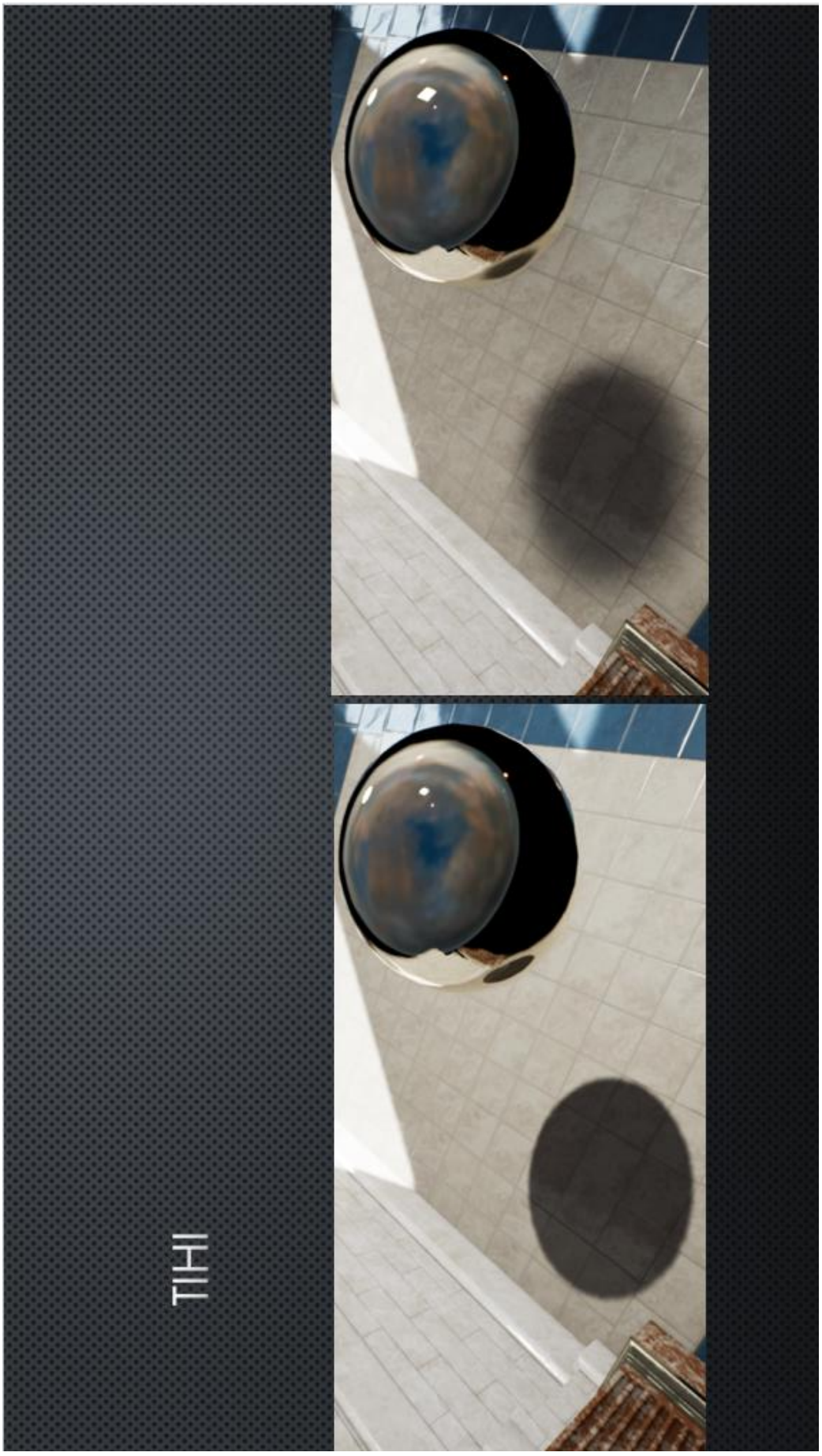

Рисунок А.8 – Тіні

Рисунок А.9 – Швидкодія тінів

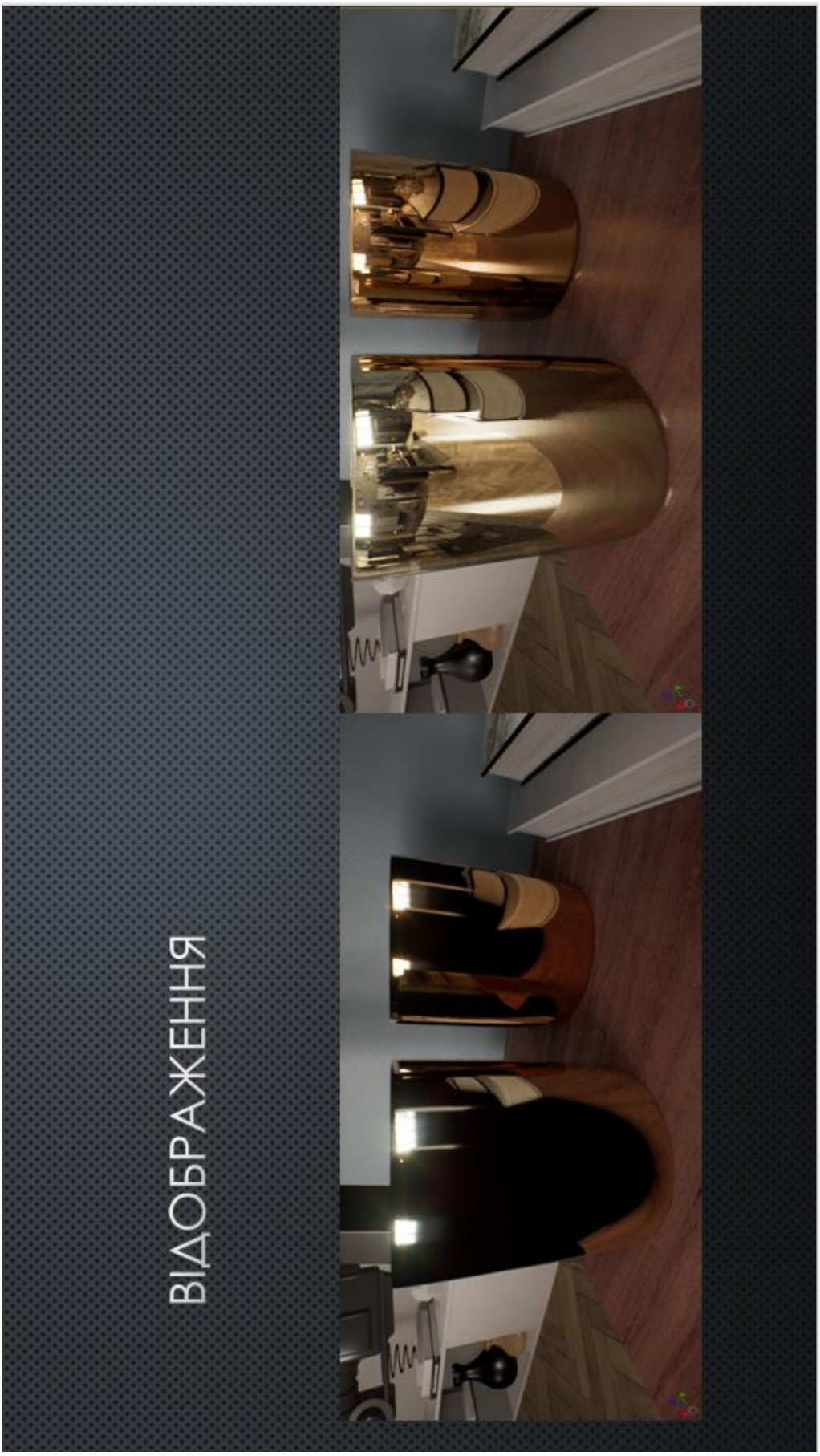

Рисунок А.10 – Відображення

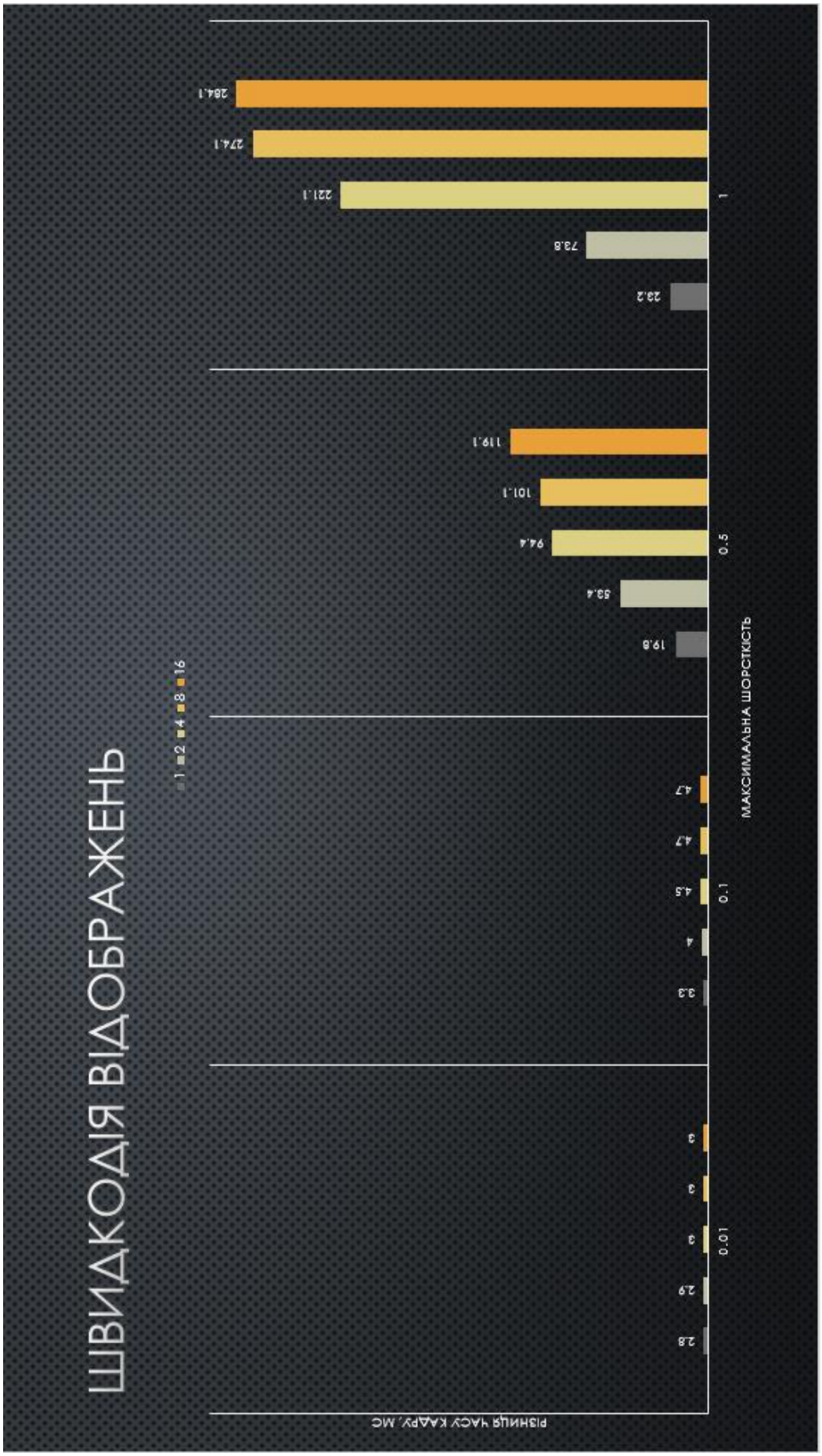

Рисунок А.11 – Швидкодія відображень

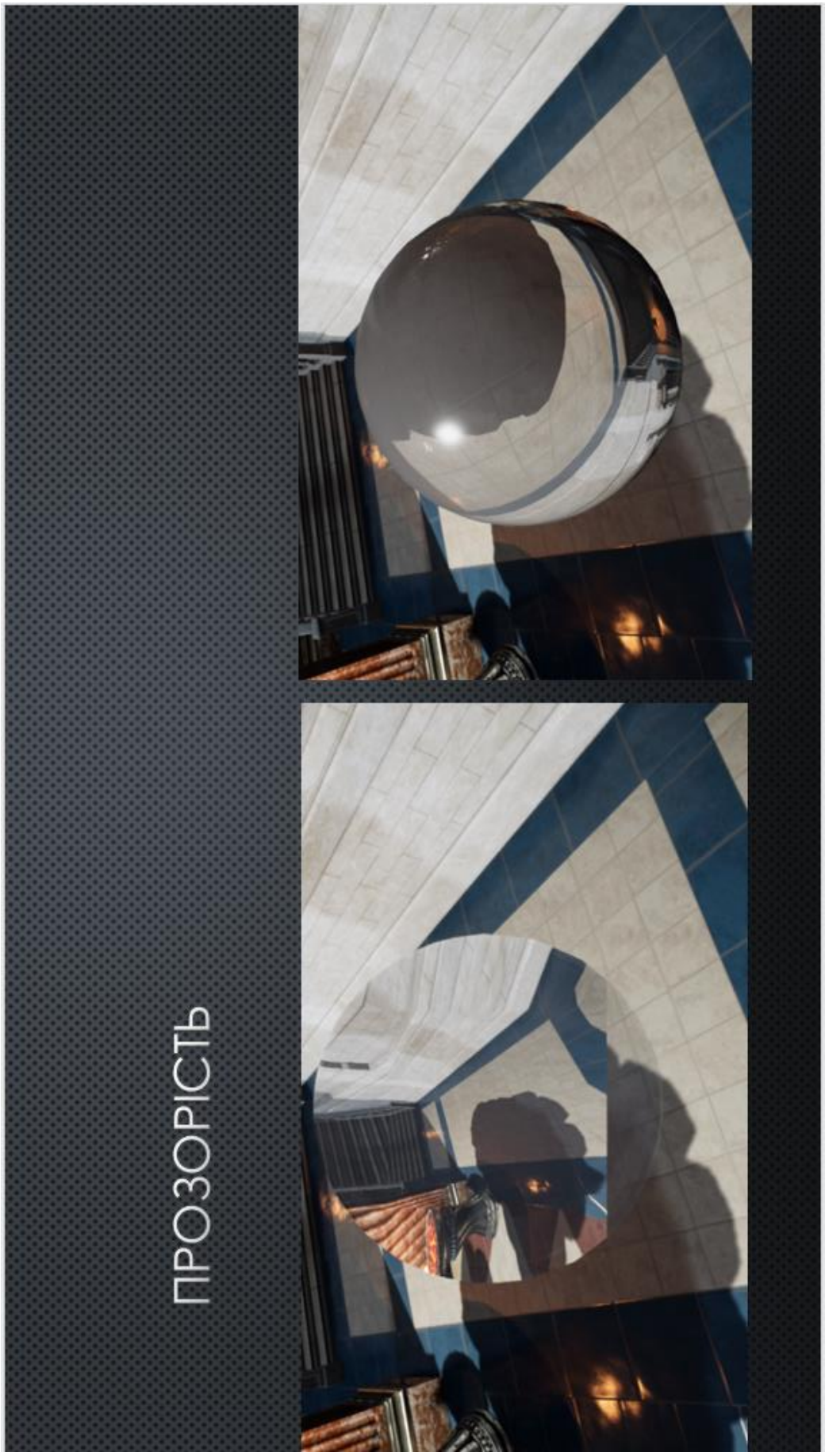

Рисунок А.12 – Прозорість

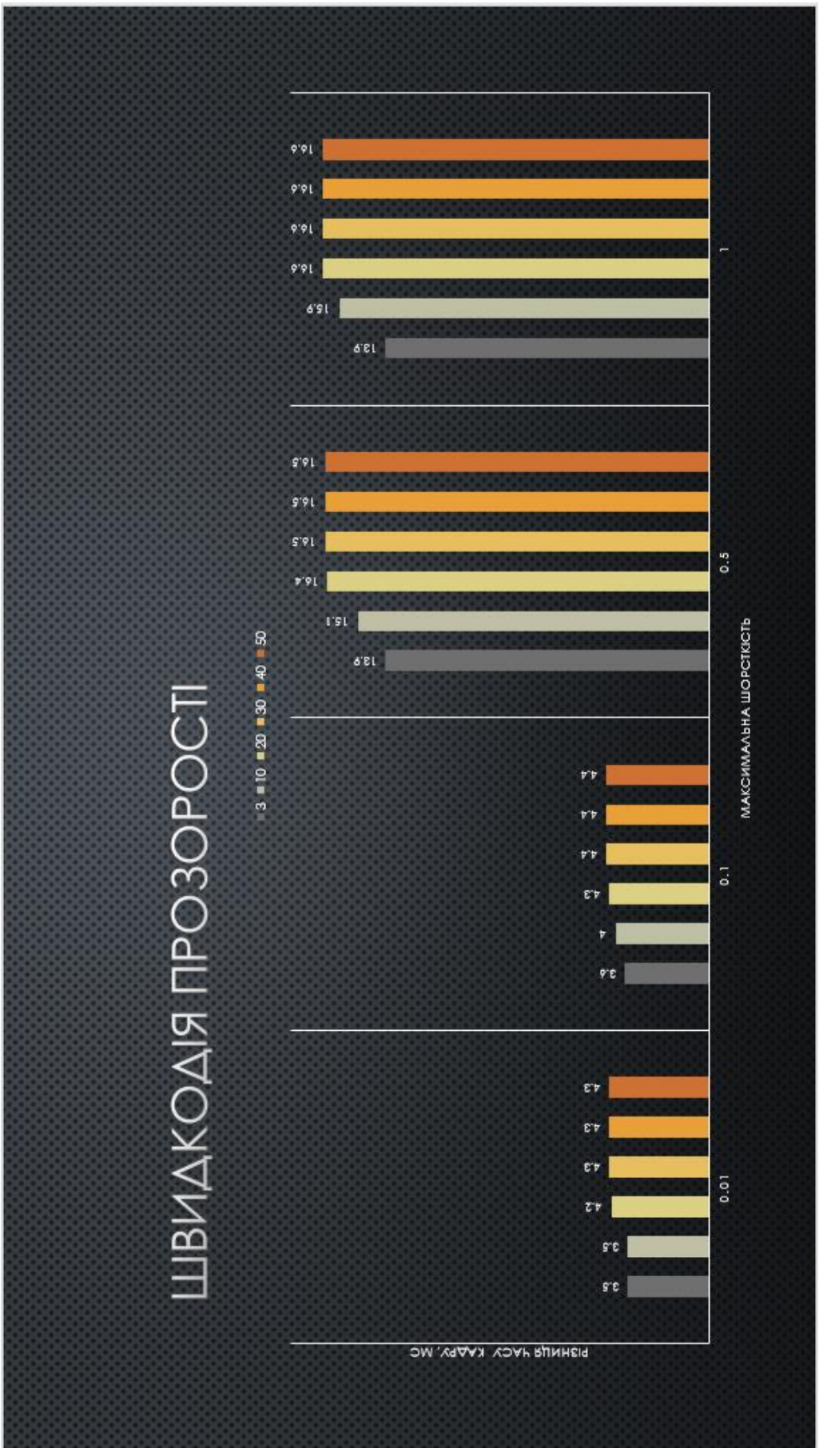

Рисунок А.13 – Швидкодія прозорості

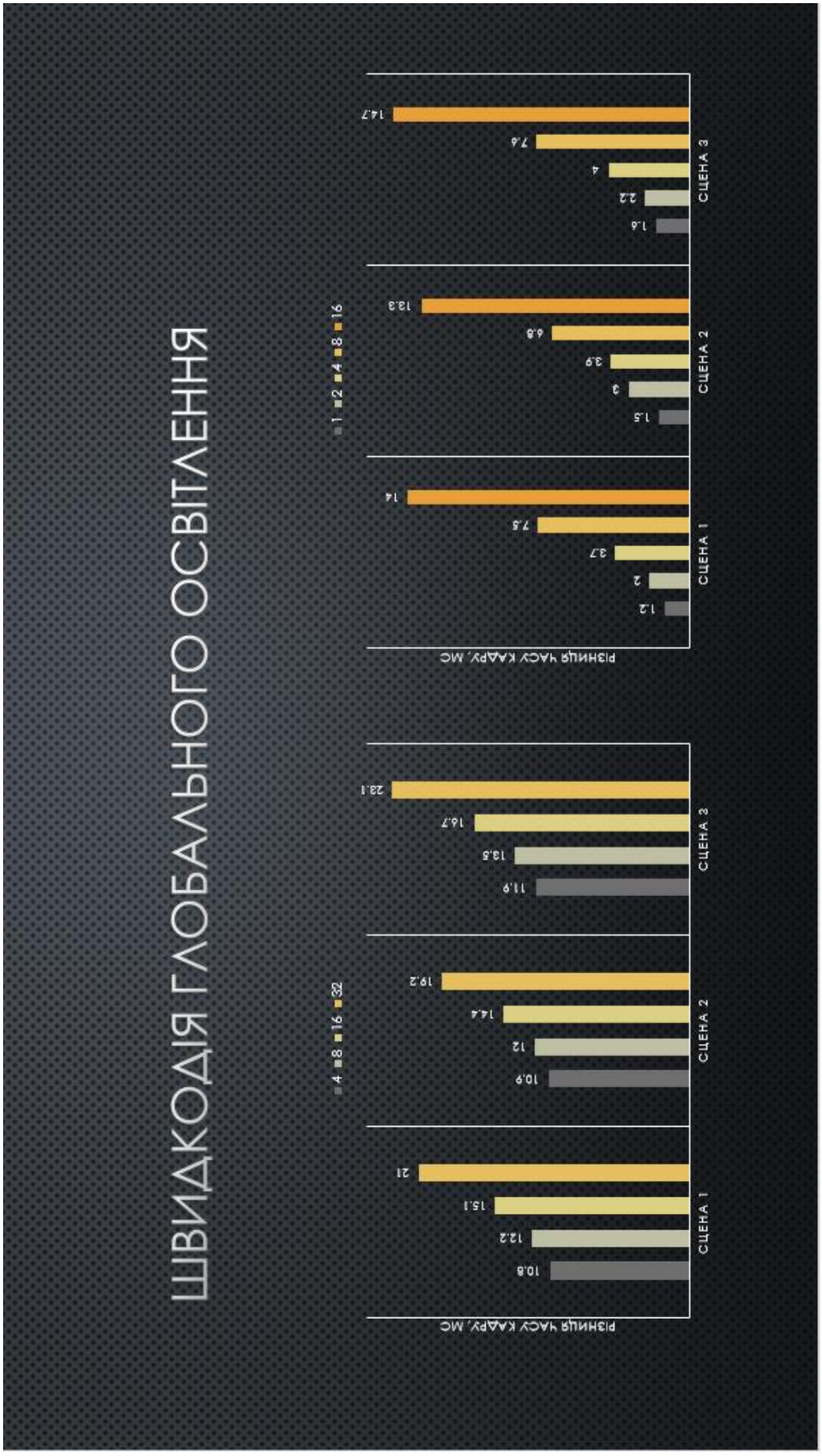

Рисунок А.14 – Швидкодія глобального освітлення

Рисунок А.15 – Подяка