虚幻引擎

图片由 Lune Rouge Entertainment 提供

nDisplay 技术

实时内容的无限制缩放

目录

简介

所有需要在大型屏幕上显示实时渲染内容的行业都面临着一个普遍挑战，即如何在各类显示媒介上缩放并同步这些内容。迄今为止，鲜少有人在这一领域取得突破，原因就在于硬件缺乏足够的处理能力，不同专用系统之间存在通信障碍，以及人们对于高帧率渲染实时内容的需求。

为了解决这些困扰全行业的难题，Epic Games 对可缩放实时内容进行了研究，并考虑了若干种解决方案，最终 nDisplay 系统应运而生。该系统与虚幻引擎集于一体，能将 3D 实时渲染内容同时呈现在多个显示媒介上。

在本文中，我们将深入探究 nDisplay 技术的幕后设计与研发工作，了解这些系统特性的研发过程以及它们如何帮助我们解决实际问题。我们将探讨当前的可用技术、时下面临的瓶颈以及未来的研发计划。

图片由 Wolf + Rothstein 提供

背景

自计算机图形学被引入媒体行业以来，就不断有呼声要求在提高分辨率和降低处理时间的基础上实现缩放渲染画面的能力。其中一种办法就是将渲染任务分配给由多台计算机组成的网络群集（即"渲染农场"）处理，或者交由多个处理器核心和软件线程处理。

随后，属于游戏和实时渲染的时代降临。实时渲染必须以 16 毫秒每帧的速度进行渲染，才能保证至少 60 帧每秒的播放速度，即公认的能够营造出真实感的速度。专用图形处理器（GPU）经历了很长一段时间的研发才实现了这一目标。

然而，此类研发往往侧重于通过增加处理器核心的数量和可用显存的容量来提升整体渲染能力，同时提高这些硬件的运行速度。诚然，对于那些相对成熟的特性而言， GPU 的结构越复杂，越有助于提升画面品质，但却无法实现以更高的分辨率在多台屏幕上同时渲染内容。此外，基于多核 GPU 的解决方案也有自身的局限性：一旦涉及多台计算机，它们就无法正确分配并缩放实时渲染内容。

对于非游戏类企业而言，如果希望将内容投射到高度复杂的显示系统上，就需要跳出游戏技术，寻求其他的解决方案。与预渲染内容相关的投射方案（例如视频播放应用）上市已久，所以十分成熟可靠。然而，实时渲染内容在这方面的进展却困难重重。迄今为止，仍未有解决方案能够高效地将实时渲染内容缩放为任意尺寸。

显示设备往往采用各种尺寸和曲率的屏幕，在文件格式和分辨率上往往也是五花八门，而这类难题不过是人们在尝试缩放实时渲染内容时遭遇的冰山一角。此外，由于开发商在为实时渲染显示设备开发内容时，会不断将视觉效果的复杂程度和真实性提升到新的高度，因此硬件需要不断升级以满足新的需求。

图片由 Reynaers Aluminium 提供

显示系统的类型

在深入了解人们对缩放渲染画面的需求之前，让我们先回顾一下相关的硬件知识和使用案例。

视频内容或实时内容的显示方案主要有两种形式：

- 借助显示屏或 LED 屏幕播放。图像数据通常通过电缆传输到一个或多个处理器上，然后通过处理器控制输出的画面
- 借助投影设备播放。多台数字投影仪将画面投射在任意表面上

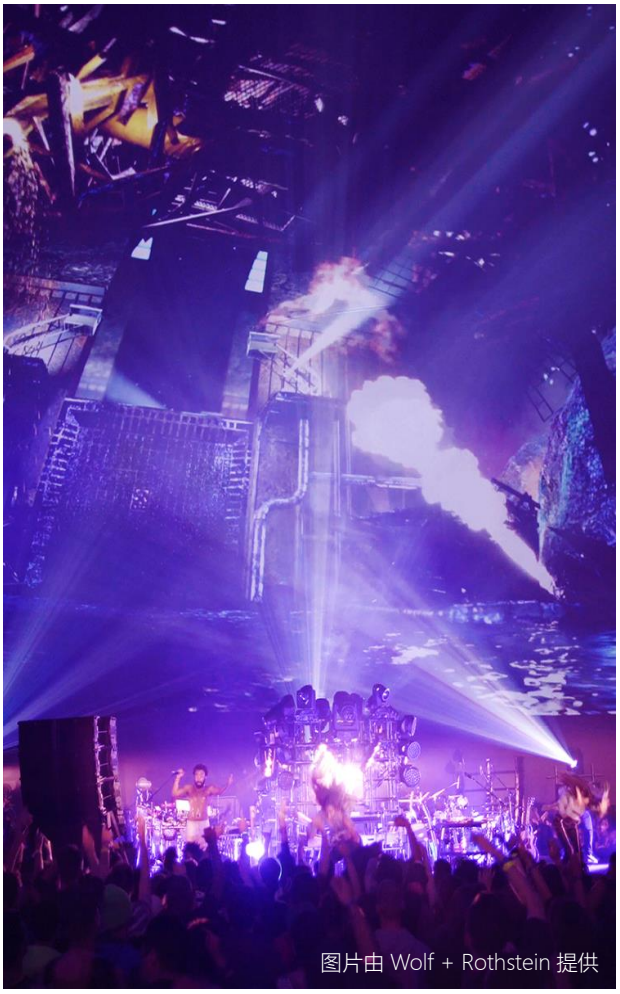
显示系统无论采用哪种拓扑结构，最终都可以归为以下一种类型：

- LED 显示屏矩阵
- 超大型 LED 屏幕（曲面或平面）
- 平面投影
- 穹顶投影或曲面投影
- 基于洞穴式自动虚拟环境（CAVE）投影的多面沉浸式场景
- 由以上两种或两种以上方案组成的复合显示方案

这些显示设备可能还需具备立体视觉功能，或需具备同步和跟踪功能，以便在 3D 空间中准确地反映用户注视点的位置。

使用高端复合显示方案的行业包括：

- 虚拟制片——用于在制作镜头内特效时使用投影或 LED 画面代替绿幕
- 建筑和制造业——用于设计审查的 CAVE 或墙幕屏（powerwall）[1]
- 仿真训练——倾斜墙幕或曲面屏幕，用于增强使用者的沉浸感
- 娱乐行业——天文馆和主题公园娱乐设施中的穹顶式投影
- 现场活动和永久性设备——由投影或 LED 屏幕组成的显示设备，拥有超大尺寸和超高分辨率，需要大量服务器来驱动内容模拟和内容播放



[1] *墙幕屏*是一种大型内容显示表面，它可以是 LED 屏幕，也可以是投影墙幕。它能以足够高的分辨率显示画面，用户即便近距离观看屏幕，也能看到丰富的细节。墙幕屏通常用于协作式应用，例如建筑或工程行业中的设计审查，或安装在博物馆这类允许观众近距离观看内容的设施中。由于这类高分辨率图像需要大量数据，所以会使用多台计算机控制投影墙系统。投影式墙幕屏需要用到多台投影仪，而由 LED 屏幕组成的墙幕屏则会包括单台或多台 LED 屏幕。

使用案例

为帮助你了解背景，我们在这里罗列了一些与实时内容缩放显示有关的使用案例。Epic 在 PixelaLabs 的协助下，帮助合作伙伴实现了他们的目标。PixelaLabs 是一家由渲染、VR 和 CAVE 技术专家组成的团队，旨在为大型项目或自定义项目提供 nDisplay 集成服务。

现场活动/穹顶投影

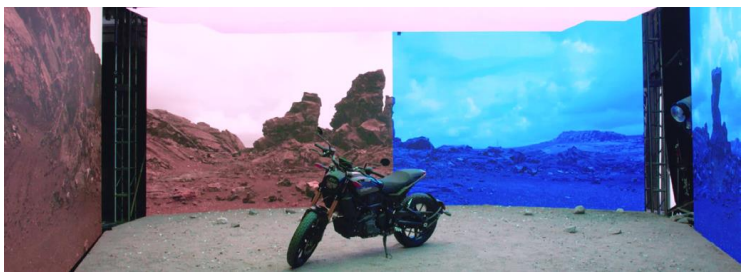
2018 年，说唱歌手幼稚冈比诺（Childish Gambino）举办了一场“灯塔”演唱会，演唱会在一个矩形穹顶下举行，穹顶上则投射有实时渲染内容。该团队使用五台计算机渲染了一张 5.4K×5.4K 分辨率的图像，然后将图像拆分到鱼眼镜头中，再发送至 12 台投影仪。

现场活动/复合显示

PY1 是一座高达 81 英尺的金字塔形场馆，可用于举办各类娱乐活动。投影系统使用了 32 台投影仪，场地的租赁设备包括激光、动感舞台元素和特效设备。

基于 CAVE/建筑可视化的仿真训练

Reynaers Aluminium 在其位于比利时迪弗尔的总部安装了 AVALON 系统，这是一个由五面墙幕围成的 CAVE 系统，该系统主要用于建筑设计演示和窗户安装培训。用户佩戴上主动式 VR 眼镜后，就能以自然的视角观看 3D 图像。AVALON 系统使用 25 台投影仪和 14 台工作站。



虚拟制片/平板 LED

Lux Machina 搭建了一座由 LED 组成的布景，其中包括四块 LED 屏幕（三面墙幕和一个天花板），它们会围住演员和道具，为他们打光并提供反射灯光。通过摄像机镜头观察时，布景中的现实元素能够无缝集成到实时 CG 场景中。

技术注意事项

在我们开始了解实时内容的缩放挑战之前，先来仔细了解下各种技术相关的事项。

显示机制

大型组网显示屏在显示实时内容时，每块屏幕都会单独显示一帧画面的一个区域。在考虑如何整合这些单独区域并让他们形成一幅统一连贯的画面时，就必须先考虑显示器的尺寸和形状以及底层的显示技术。

投影屏幕种类

在投影类设置中，图像会通过投影仪投射到平面上。投影仪投射的画面会与相邻投影仪投射的画面产生部分重叠，以便实现平滑的混合效果。重叠区域的大小通常约为投射画面的 15%到 20%。显示实时内容的投影屏幕通常有以下几种类型：

- **平面或曲面屏幕**——投影仪会并排放置或垂直叠放。如果投影区域较大或亮度要求较高，投影仪还可能同时并排和垂直放置。投影内容会在重叠区域发生混合和重叠。
- **球形或金字塔形屏幕**——由多台投影仪组成的复杂阵列会覆盖整个表面。通常，为了增加亮度会出现明显的重叠区域。
- **任意形状的屏幕**——实际上任何形状的表面都可用于投影，只要图像能以足够亮度显示。

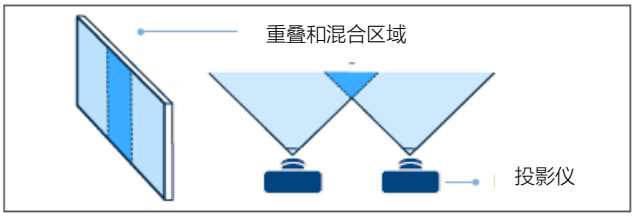


图 1：平面投影设置

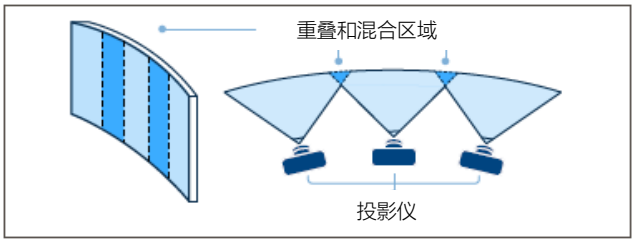


图 2：曲面投影设置

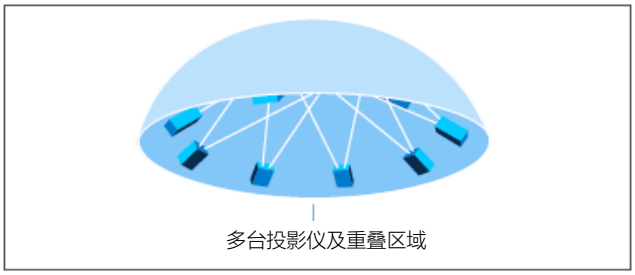


图 3：球面或穹顶投影设置

图 4：平面、曲面和球面投影设置示例。【图片由 Scalable Display Technologies 提供】

图 5：曲面 LED 屏设置示例。【图片由 Moment Factory 提供】

LED 屏幕

就在不久前，所有 LED 屏幕还都是平面，所谓的“曲面”屏幕无非是多台平面屏幕略微倾斜并排放置后形成的效果。但如今，LED 屏幕几乎已经可以被设计成任何形式、形状、分辨率或像素间距。平面 LED 屏幕还能够通过复杂的排列模式形成三维显示效果。

由于 LED 屏幕的图像数据通过电缆传输而非投影仪，不同屏幕图像之间的接缝可以精准对齐，因此屏幕之间的图像无需重叠或混合。

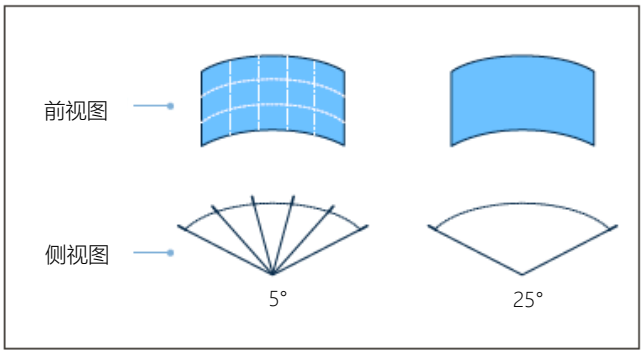


图 6：由多台平面屏幕组成的曲面屏幕（左）和单块曲面屏幕（右）

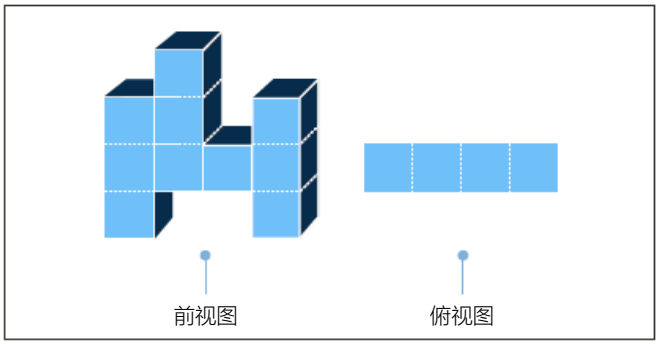


图 7：由多块平面 LED 屏幕构成的 3D 显示设备

同步

对于能否使用多块屏幕组成大屏幕来显示实时内容，同步无疑是一项决定性因素。在一个由多个子系统构成的渲染主系统中，所有子系统都必须严格遵循计时规定（精确到毫秒），以产生画面无缝同步的错觉。

采用多台显示器的投影设置通常需要在软件和硬件层面上都具备同步功能。不仅要让所有计算机都通过相同的模拟计时信息同时生成内容，而且还要让画面交换（显卡缓冲区中当前图像和下一帧图像的交换操作）在正确的时间发生，以防画面出现“撕裂”。

对于 VR 和其他类型的立体显示来说，同步则会加倍复杂，因为两只眼睛看到的两幅画面都必须确保完美协调。这里我们会讨论与 nDisplay 最密切相关的同步领域。如需深入了解这些功能的实施，请查看虚幻引擎官方文档中的“nDisplay 中的同步机制”。

确定性

在管理同步上，有两种方法：

- **确定性方法：**每台服务器（即计算机或渲染节点）在设置时，只要给定一组特定输入，就能始终输出可预测的结果。这就意味着服务器只需要精确的时间信息以及各台设备的输入/输出信息，就能与系统中的其他服务器进行同步。
- **非确定性方法：**为确保同步，系统会强制为场景中所有 Actor 或对象复制变换矩阵以及其他相关属性，然后在整个系统中同步这些信息。

两种方法各有利弊。确定性系统的主要优点在于项目相对简单，而且由于不必在每一帧画面中同步所有对象的变换数据，所以能最大程度节省数据带宽。而缺点是，如果某个系统出现偏差，随着时间的推移，这种偏差会引发未知的问题。渲染一致性可能会受到严重影响，导致画面不连贯并出现瑕疵。

硬件同步和同步锁定

尽管 nDisplay 系统中的主计算机可以确保群集中的所有群集节点计算机获取游戏玩法相关的时间信息（例如需要渲染哪一帧画面），但是仍然需要使用专门的硬件同步卡和具备兼容能力的专业显卡，以便在物理显示设备上同时同步渲染画面。

例如，在广播应用中往往需要同步大量设备，例如摄像机、监视器和其他显示器，让它们能够在同一时间精准无误地切换并显示下一帧画面。在该行业中，同步锁定被广泛应用。

通常，设置中会包含一个硬件生成器，它负责将时钟信号发送给需要同步的硬件。就用于实时渲染的计算机而言，它们所使用的英伟达 Quadro 系列专业显卡和 Quadro Sync II 显卡都能支持这种技术，后者还能锁定接收到的定时信号或脉冲。

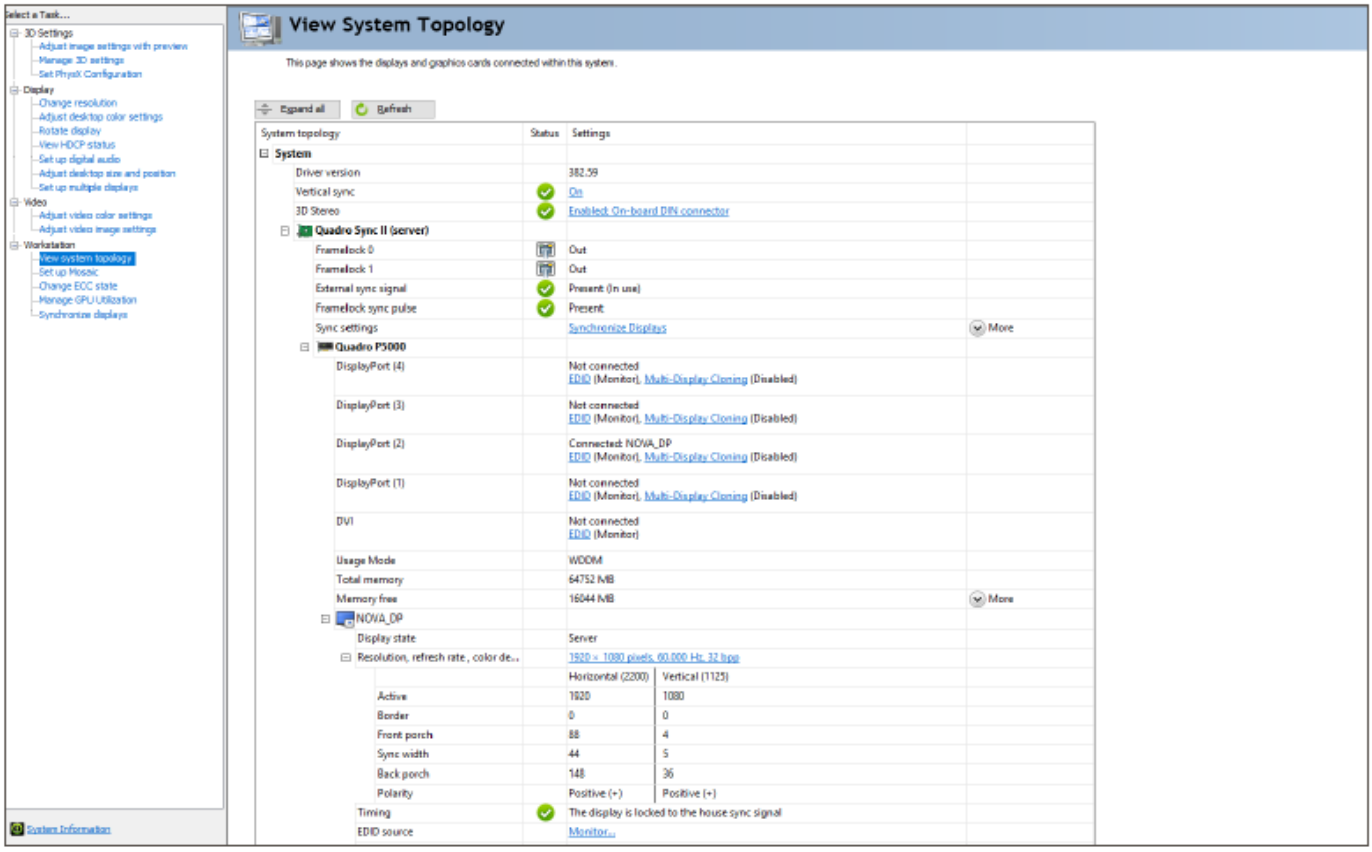


图 8：英伟达控制面板截图——同步锁定的查看系统拓扑结构

菊花链与直接同步锁定

菊花链是一种信号锁定技术，可配合直接同步锁定。在直接同步锁定中，主时钟信号会被发送给单台计算机或设备（本示例中是主计算机），然后通过单独的电缆将信号传播给所有其他计算机。

根据之前使用 nDisplay 的经验，我们发现直接同步锁定（所有设备直接获取主设备的时钟信号）比菊花链更简单、更有效。但是随着基于菊花链的新硬件渠道的推出（英伟达 Swap Sync/Lock），以及虚幻引擎 4.25 提供的对于信号锁定的替代方案，使得菊花链可能更加可靠、更具成本效益。

同步测试

测试缩放显示的同步效果可能会十分棘手，因为很多情况都可能导致同步失效，例如：

- 由于错误的时间戳导致模拟错误的帧
- 显示设备的计时功能关闭

为了测试同步，我们创建了一个简单的测试用项目。在这个项目中，我们让单个物体快速穿过整副屏幕画面。如果系统能够完全同步，物体在穿越屏幕边缘时会维持正确的形状。否则，物体在穿越屏幕边缘时会出现画面撕裂。

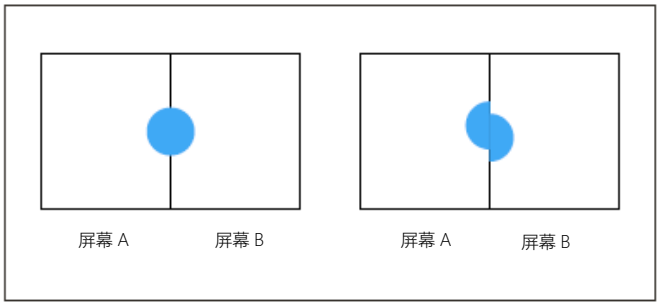


图 9：用于同步测试的简单场景。左侧是正确同步的效果，右侧是未能同步的效果。

后期特效

例如泛光、镜头眩光和动态模糊这类后期特效都需要在屏幕空间中计算，这意味着它们只能在渲染完整帧画面后才能应用。这是因为这类特效通常不会局限于图像中的某一区域。它们需要邻近区域的像素信息才能获得正确的渲染效果。

例如，仅仅是泛光这一个特效通常就会涉及整幅画面。在使用确定性方法的分布式渲染系统中，源于图像中某一区域的泛光特效不会“到达”邻近区域，而且泛光区域和非泛光区域之间的过渡会非常生硬。

在缩放画面时，这类特效应该被禁用，以免混合区域出现视觉瑕疵。要使用这类特效并非不可能，但必须在屏幕边缘上小心处理，而这通常需要使用昂贵和先进的技术。

现有技术

作为开发新功能时采用的策略之一，Epic Games 将始终评估能否将现有工具作为新功能添加进虚幻引擎（UE4）中。经过大量研究，我们发现下列技术有助于我们实现缩放显示的目标。

MPCDI（多屏投影通用数据交换）

MPCDI（多屏投影通用数据交换）标准由 VESA 的多屏投影自动校准（MPAC）任务组制定。它是一种标准数据格式，用于投影校准系统与多屏配置中的设备间的通信。该标准为多屏幕投影仪系统提供了一种生成所需数据的方法。依据该标准，人们可以借助各类设备将分散的显示组件组合起来，以便输出一张无缝的完整图像。任何新硬件都能根据该标准轻松集成入系统中。

MPCDI 已广泛用于行业内的内容制作商和供应商，例如：

- Scalable Display Technologies
- VIOSO
- Dataton Watchout
- 7thSense Design

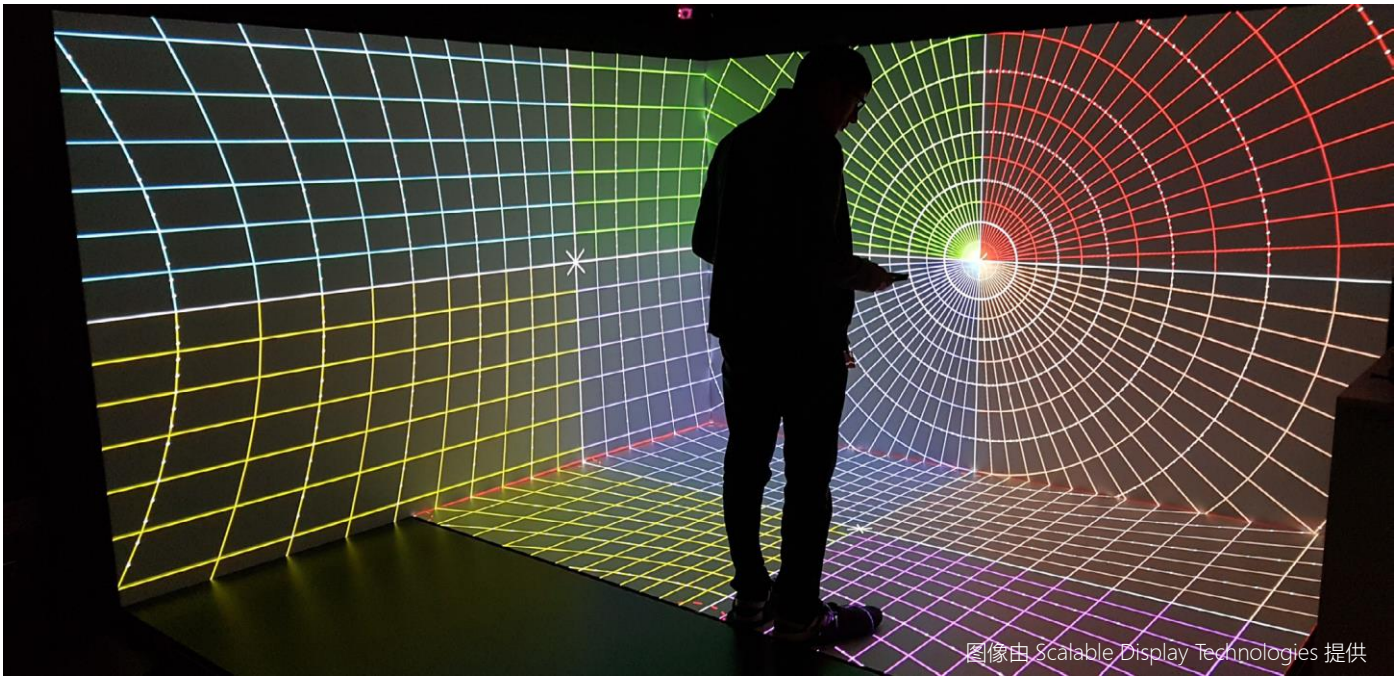
Scalable Display 的 EasyBlend 技术

Scalable Display Technologies 是一家专门为复合投影系统开发软件和 SDK 的公司。他们的 SDK 旨在提供一种解决方案，即借助扭曲和混合等手段让多台显示设备呈现单幅画面。鉴于 Scalable Display Technologies 的 EasyBlend 解决方案在处理大型图像的扭曲和混合时已经相对成熟，我们决定将它引入虚幻引擎，以帮助我们实现目标。

系统规格

我们考虑了人们在可缩放实时内容这一领域中最可能出现或最普遍的需求，并查看了现有的技术，最终认定某个系统在使用虚幻引擎的解决方案前必须能够做到以下几点：

- 能够在网络中的计算机阵列部署并启动多个虚幻引擎实例
- 确保所有相关计算机上的内容都能完全同步恢复
- 启用主动/被动立体视觉
- 管理并分配各种输入源，例如 VR 追踪系统
- 能够兼容各种配置类型的显示设备（例如，不同尺寸、不同空间朝向或不同分辨率的显示设备）


图像由 Scalable Display Technologies 提供

nDisplay 解决方案

为满足这些需求，Epic Games 开发了 nDisplay 系统。nDisplay 能够将虚幻引擎的渲染任务分配给许多联网的计算机，因此无论有多少台显示设备都能生成图像，而且能够正确实现帧/时间同步，世界空间中的视椎体能在屏幕阵列中正确显示，可视化系统中的确定性内容会始终保持一致。

从本质上说，nDisplay 技术扩展了虚幻引擎，因为它能将摄像机画面的渲染任务分配给任意数量的计算机，并在任意数量的显示设备上显示图像。

在全面考察后，我们认为在虚幻引擎中实现 nDisplay 的最佳方法就是让群集节点自动附加到虚幻引擎项目中当前活跃的摄像机位置上。虚幻引擎中的摄像机在拍摄画面后，该画面会基于 nDisplay 的设置进行扩展、渲染和分配。

nDisplay 具备以下功能：

- 在不同的虚幻引擎实例之间同步 Actor（例如摄像机、动画、粒子等）
- 充当按键、方向轴和定位的监听器
- 充当 VRPN 服务器，用于追踪 ART 和 Vicon 等设备
- 支持 DirectX 11 和 DirectX 12 图形库，无论是普通模式还是立体模式（帧序列四缓冲区、并排、顶部/底部）
- 支持英伟达的 Quadro Sync 同步功能，有助于保证帧的连贯性、灵活性和可缩放性
- 为立体系统提供非对称视锥体配置

框架

nDisplay 工具包内含一个插件、一组配置文件以及为虚幻引擎开发的应用程序。它包含以下组件：

- nDisplay 插件——在运行时使用，用于为不同实例、可能出现的 Actor 复制对象、输入管理系统提供网络接口。它可以配置渲染子系统以显示节点拓扑
- nDisplay 配置文件——描述显示系统的拓扑结构和项目设置的整体中心位置
- nDisplay 启动器和 nDisplay 监听器——用于在不同的组网计算机上启动并控制虚幻引擎的实例，总计“n”个实例。每台组网计算机都会与一台或多台显示设备连接

与虚幻引擎特性进行集成

确定性和非确定性功能

我们在前文中曾讨论了确定性系统的各种利弊。在该类系统中，由于各节点无需与其他节点共享帧信息，所以同步相对简单一些。

在虚幻引擎中，我们倾向于让游戏玩法、物理模拟和渲染功能是完全确定性的。然而，部分子系统当前还不是完全确定性的。下面这张图表展示了虚幻引擎中的各类特性的确定性情况。

特性	是否为确定性的?	备注
简单物理模拟	一定程度上采用	无论使用何种物理解算器，碰撞体积、胶囊体、平面、盒体和球体都会表现出不一致性。为确保准确性，需要对它们进行复制。
复杂物理 • 刚体 • 软体 • 布料 • 车辆和关节 • 骨架	一定程度上采用	当前 PhysX 解算器所采用的随机噪点会阻碍系统在一段时间内保持同步。为了确保精确，需要复制这类对象。
Niagara 粒子	是	我们在使用此系统时还未遇到过时间偏差。
Sequencer	是	是的，其本质上就是确定性的，能够按照预期运作。
蓝图编写的游戏逻辑	是	多数蓝图逻辑都能按照预期运行，除非蓝图包含一些已知的非确定性功能，例如使用非确定性设备或使用非确定性函数。
蓝图编写的随机逻辑	否	虚幻引擎中的随机功能会从根本上妨碍逻辑实现完全的确定性。注意，某些在蓝图中配置的粒子系统会产生随机序效果，使用时应当仔细检查。

表 1：虚幻引擎各个特性的确定性程度

我们未来的计划包括重写虚幻引擎的物理引擎，以便完全支持确定性行为。此外，我们还支持复制功能，用于在需要时强制确保画面的连贯性，但代价是牺牲额外带宽并且需要对项目进行自定义设置。

非对称视锥体

在摄像机的传统用法中，摄像机位于画面的中心，因此能提供一个对称的视锥体。

分布式系统需要对摄像机位置和自定义视锥体进行定义，然后将它们发送给各台计算机用于渲染画面。由于一台 UE4 摄像机要为多台屏幕或投影仪提供视图，因此摄像机的视锥体必须被相应地分割成多个视锥体，每个视锥体分别专门为某个投影仪或某块 LED 屏幕提供图像。从本质上说，这些视锥体都是非对称的。

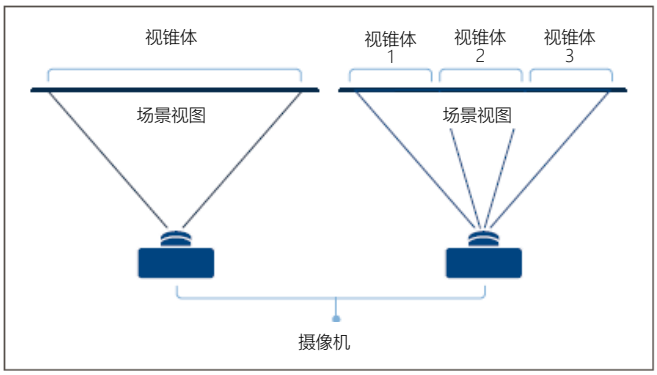


图 10：对称视锥体（左）和非对称视锥体（右）

请注意，视锥体被分割时，渲染图像的透视效果不会发生改变。分割视锥体只是为了便于对摄像机拍摄的图像进行分布式渲染。在 nDisplay 系统中，这类图像通常会偏离中心。

后期处理特效

由于屏幕的边界位置或投影仪的重叠/混合区域存在潜在的连贯性问题，所以我们建议你禁用那些已确认有可能导致画面出现瑕疵的屏幕空间后期处理特效。然而，如果项目中确实需要使用这类特效，有一种方案也许能帮你解决问题，那就是在视锥体之外渲染额外的像素。这种方法称为"超扫描"，它允许你应用此类后期特效，但渲染时间会增加。你可以通过扩展 nDisplay 来进一步开发这种方法。

以下是一些应当禁用或谨慎使用的后期处理特效，因为它们可能会导致画面撕裂或其他连贯性问题：

- 泛光
- 镜头眩光
- 自动眼部适应
- 动态模糊
- 环境光遮蔽
- 抗锯齿（虽然很不明显，但某些技术条件下可能会显示出不同）
- 屏幕空间反射
- 暗角效果
- 色差

当你在考虑是否应该在屏幕阵列中添加屏幕空间特效时，你需要权衡一下，你付出额外精力去实现的特效是否能够给观看的用户带来满足感（并且你要考虑到特效最后可能仍然无法正常工作，导致不但没有增强体验，反而削弱了体验）。为项目制定这类决策时所涉及的许多方面已经超出了本文的讨论范围。

MPCDI（多屏投影通用数据交换）

由于 nDisplay 支持 MPCDI 标准，所以它能以标准化和规范化的方式对描述复合投影仪系统的数据进行读写，这样我们就能轻松地与业内各种其他工具进行通信。

由于 MPCDI 刚刚推出不久，所以它在使用上和用户体验上还仍有欠缺。为了克服这类局限性，我们正在开发一款解决方案，可用于在虚幻编辑器中实时预览 MPCDI 文件的数据。

目前，用户能够以 MPCDI 文件生成的网格体数据为基础，为物理显示设备生成程序化的网格体。

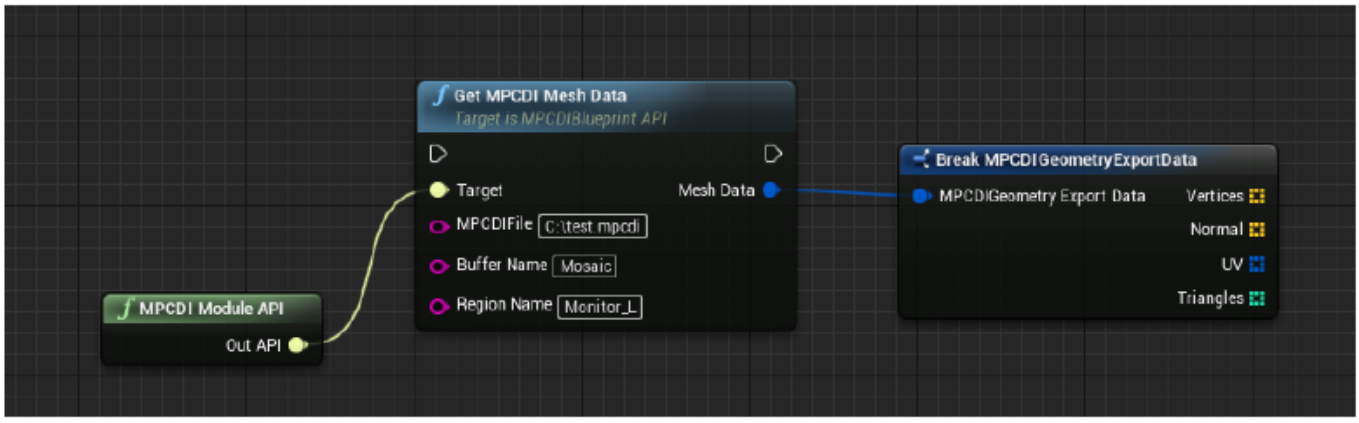


图 11：蓝图中的 MPCDI 设置示例

在撰写本文时，市面上尚未推出用于实现 MPCDI 的工具，所以 MPCDI 配置文件的生成和操作依赖于底层的软件公司。将来，我们可能会在 nDisplay 中引入一个用于显示或编辑 MPCDI 配置文件的工具，但目前而言，我们尚无法实现该功能。

可缩放显示——EasyBlend 的集成

nDisplay 支持通过集成 Scalable SDK 和 EasyBlend 这类行业标准中间件来实现弯曲和混合效果，并且可用于所有受支持的模式、MPCDI 的原生弯曲和混合效果以及自定义实现。

我们集成了 EasyBlend，以便在配置复合投影系统时提供一种无缝体验。用户在使用第三方工具或软件完成校准后，只需在 nDisplay 的配置文件中指定一些参数即可开始运行。

局限性

在设置和使用 nDisplay 时需要手动完成以下步骤：

- **配置：**手动设置配置文件，定义显示设备的拓扑结构、投影策略、视口、充当渲染节点的计算机、追踪设备和系统所有其他组件。
- **调整项目：**在对 nDisplay 进行一系列更新后，你只需对项目稍加调整，就能避免它使用不兼容的或非确定性的功能。本质上，只要是线性功能、动画功能、不依赖复杂物理机制的功能、随机函数或屏幕空间特效，都能按照原样工作。在某些更为复杂的项目中，可能需要复制部分 Actor 才能确保正确同步。
- **部署：**借助手动方式或自定义工具，对你的项目文件和资源进行配置并复制到目标渲染计算机上。然后使用我们提供的第三方工具远程启动你的 nDisplay 项目。

以下是撰写本文时 nDisplay 已知的一些局限性：

- nDisplay 目前只能在 Windows 7 及更高版本的 Windows 系统上运行（支持 Directx11 和 Directx12）。
- 四缓冲（主动）立体功能只支持在 Windows 8.1 及更高版本的 Windows 系统上运行。
- 不再支持 OpenGL。
- nDisplay 目前不支持 Linux 系统。
- 对于帧锁定和同步锁定的支持只适用于英伟达 Quadro 这类专业显卡。
- nDisplay 不支持自动曝光、泛光、平面反射以及部分着色器效果。你仍然可以使用它们，但你可能会在显示的内容中发现画面发生不连贯或出现瑕疵。
- 不支持对二维 UMG 界面进行操作（悬浮、点击和视口对齐）。你可以通过 OSC、REST 或其他可用的远程控制协议来制定替代方案，以便控制 nDisplay 系统。

注意，nDisplay 可以在 GPU 渲染层面（实时内容的主要瓶颈）上进行缩放，但它无法缩放 CPU 相关的运算，例如游戏逻辑和物理检测。所有群集计算机仍需单独处理所有基于 CPU 的运算。换句话说，由于 nDisplay 仅仅是一种分布式渲染系统，所以在 CPU 层面上，它无法提供任何加速效果。

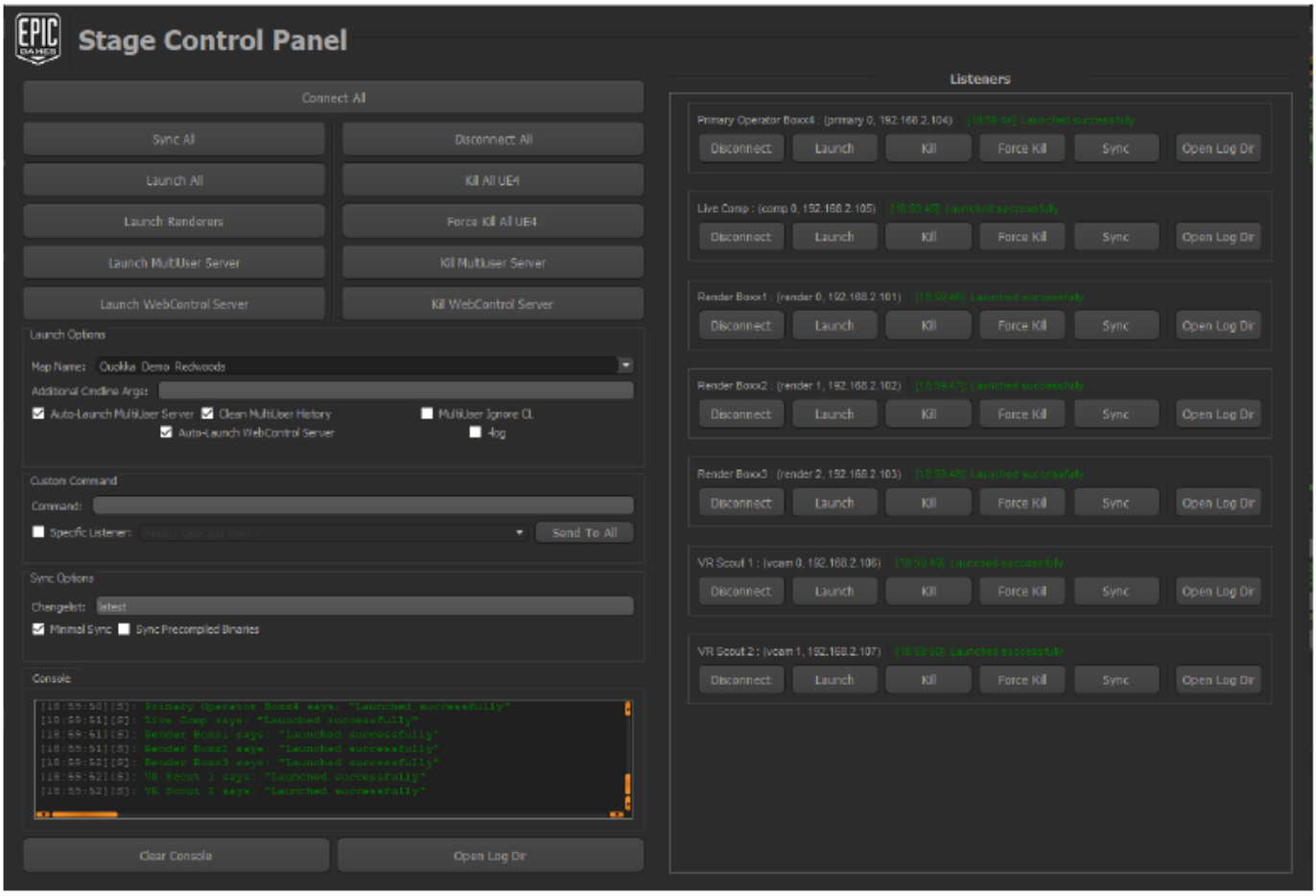


图 12：自定义部署工具示例

部署

在撰写本文时，nDisplay 尚未提供任何自动化内容工具或项目部署工具。这是由于大多数用户使用的项目都极为复杂，且涉及一些十分特殊的网络限定条件，比如防火墙策略，所以他们通常喜欢用自己的方法来管理并部署/复制内容。

然而我们打算今后添加一组最低限度的快速部署功能，帮助你测试 nDisplay，从而进一步减少缩放虚幻引擎渲染内容所需的手动操作。

与此同时，我们的内部团队已经提出了一些类似于图 12 所示概念的工具。我们会以此为灵感，进一步开发未来版本的部署工具。

nDisplay 工作流程

下图显示了 nDisplay 如何在网络中与显示设备一起工作。

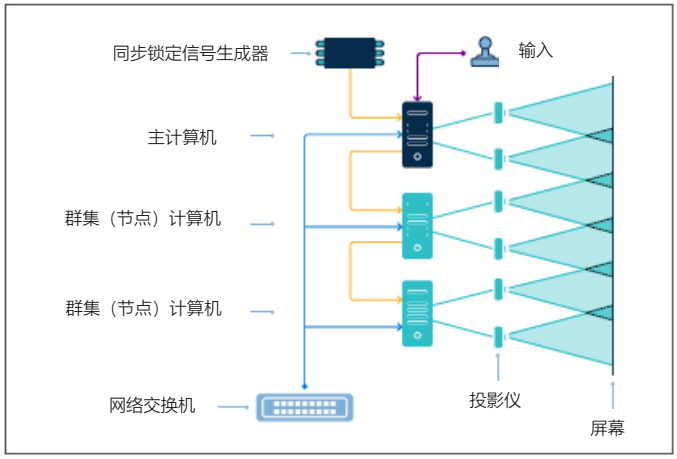


图 13：投影显示设备的组网结构

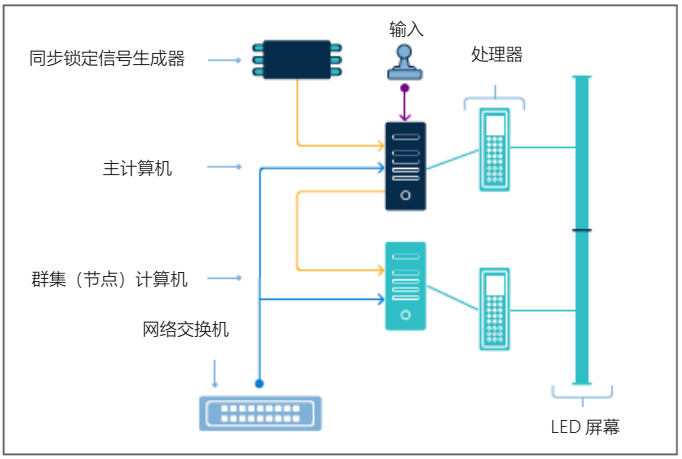


图 14：LED 屏幕的组网结构

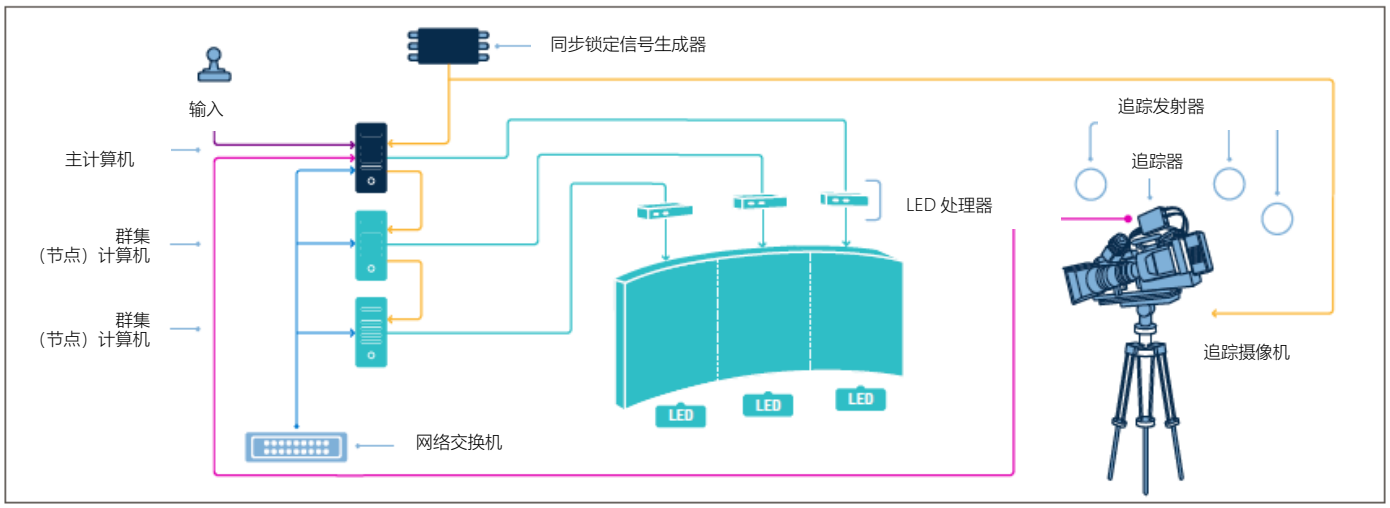


图 15：投影系统（带有追踪摄像机）的组网结构

nDisplay 插件可以在现有项目中启用，也可以通过 nDisplay 模板在新建项目时自动启用。有关详细信息，请参阅 nDisplay 快速上手文档。

主计算机

主计算机相当于 nDisplay 的调度器。它负责集中处理以下内容：

- 以同步方式管理数据，例如输入数据、摄像机追踪数据以及自定义群集事件，并将这些数据分发给其它位于 nDisplay 群集网络中的群集（节点）计算机
- 确保所有计算机同时接收并确认相同的输入及数据
- 管理跨计算机群集的计时信息
- 对可能出现的 Actor 及数据的复制内容进行管理并分发给其它计算机

群集（节点）计算机

- 有效执行与主计算机相同的游戏逻辑和物理模拟
- 在与主计算机保持同步的情况下，渲染额外的摄像机视椎体
- 与所有群集计算机保持帧锁定和同步锁定

配置文件

配置文件位于 nDisplay 功能的根目录中。它本质上描述了计算机和显示机制的硬件设置方式以及两者之间的关系。我们的 nDisplay 配置文件参考文档已经对 nDisplay 配置文件的设置方式进行了详细描述，但在本节中，我们将再详细介绍一些与 nDisplay 有关的概念和术语。

窗口、视口和屏幕

如需配置 nDisplay 系统，请务必先理解一些在 nDisplay 语境中具有特定含义的关键术语。

窗口——整幅帧画面的其中一块区域，由群集中的单个节点（通常是一台计算机）渲染/显示。

视口——窗口的其中一块矩形区域。例如，窗口中的图像可能由四个视口组成，每个视口都会单独渲染，然后组合成窗口。nDisplay 在配置中会指明窗口由多少个视口组成，并且会给出这些视口在窗口中的位置信息。请注意，你完全可以只设置一个视口来显示窗口。

屏幕——显示设备在现实世界中的物理位置、尺寸和方向。在投影系统中，屏幕是一个可以确定摄像机视锥体的矩形区域。

摄像机——虚幻引擎场景中的摄像机进行偏移后得到的结果。该偏移有助于对图像进行微调，以提升观看者的体验。

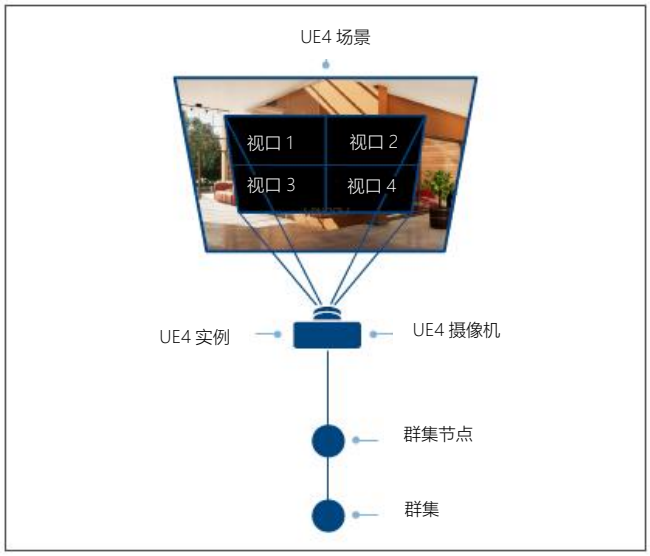


图 16：窗口、视口、实例、摄像机和群集节点在采用单个群集节点的 nDisplay 系统中的相互关系。

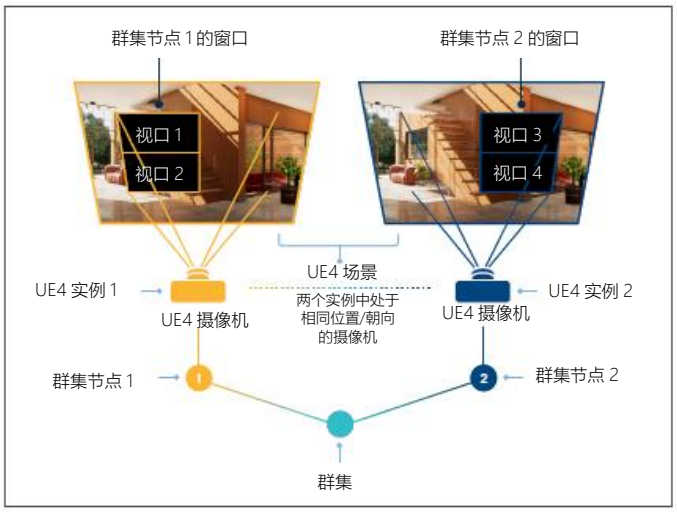


图 17：窗口、视口、实例、摄像机和群集节点在采用多个群集节点的 nDisplay 系统中的相互关系。

以下描述有助于解释这些元素之间的逻辑关系：

- 一个群集拥有一个或多个群集节点。
- 一个群集节点只能对应一台 UE4 摄像机，但是一台 UE4 摄像机可以与多个群集节点关联。
- 一个群集节点只能显示一个窗口，这个窗口可以显示部分或完整的 UE4 摄像机视图。
- 一个窗口可以由一个或多个视口组成。
- 每个视口都有一个投影策略，该视口可显示部分（或完整）画面，并且该画面由 nDisplay 配置文件中的其中一台可用摄像机拍摄。多个视口可以使用相同的投影策略。
- 视口不可重叠。
- 摄像机可以被不同的视口重复使用。

群集和群集节点

群集节点用于描述 nDisplay 的计算机网络配置，指定哪台计算机作为主计算机，并且指定每台计算机分别会被分配到哪些窗口。群集在创建时会与虚幻引擎场景中的当前摄像机挂钩，摄像机在场景中拍摄的画面最终会被分配给不同计算机进行渲染。

nDisplay 本质上是在当前的虚幻引擎摄像机中添加额外的视点。你可以把它想象成额外多了一双眼睛，或者更准确地说，是多了一个可以在任意位置观察你所在场景的虚拟视口，只不过连接到的是主摄像机的路径。如果你在虚幻引擎的场景中将摄像机向前移动 5 个单位，那么整个 nDisplay 群集都会朝相同方向移动 5 个单位。

一个群集节点与一个 UE4 应用程序实例相关联。通常应用程序实例都会在各自的计算机上运行，但是你也可以让多个实例在同一台计算机上运行，让群集节点分别与各个实例相关联。

群集由群集节点组成。在那些最为常见的设置中，比如用单台显示设备显示摄像机画面时，只需用到一个群集。在一些比较少见的设置中，会让两台或多台显示设备显示不同的 UE4 场景（或者在同一个场景中以不同的摄像机角度显示画面）。在这类情况中，你需要为每一个 UE4 场景设置单独的群集。

投影策略

投影策略是一种抽象的描述性内容，用于明确将投影的输入数据发送至何处以及如何计算输出。这意味着每种策略都可能拥有其独有的属性，并且只有它知道如何解释并利用这些属性。

nDisplay 支持多种策略。以下是一些最为常用的策略：

- **简单策略**——一种用于在常规显示设备上显示画面的标准策略。
- **EasyBlend 策略**——由 Scalable SDK 集成了 EasyBlend 校准数据，支持弯曲/混合/梯形畸变功能。适合多台投影仪在非平面和复杂显示表面上投射画面，例如曲面或穹顶形状的表面。
- **MPCDI 策略**——集成了 MPCDI 标准，用于那些以行业协议为基础的复杂类型项目。

配置文件示例

采用 EasyBlend 策略的完整配置文件示例

```
[info] version="23"
[cluster_node] id="node_left" addr="10.1.100.2" window="wnd_left" master="true"
[cluster_node] id="node_right" addr="10.1.100.3" window="wnd_right"
[window] id="wnd_left" fullscreen="true" viewports="vp_1,vp_2"
[window] id="wnd_right" fullscreen="true" viewports="vp_3,vp_4"
[projection] id=proj_easyblend_1 type="easyblend" file="C:\Program Files\Scalable Display\DEI\LocalCalibration\ScalableData.pol" origin=easyblend_origin scale=1
[projection] id=proj_easyblend_2 type="easyblend" file="C:\Program Files\Scalable Display\DEI\LocalCalibration\ScalableData.pol_1" origin=easyblend_origin scale=1
[projection] id=proj_easyblend_3 type="easyblend" file="C:\Program Files\Scalable Display\DEI\LocalCalibration\ScalableData.pol" origin=easyblend_origin scale=1
[projection] id=proj_easyblend_4 type="easyblend" file="C:\Program Files\Scalable Display\DEI\LocalCalibration\ScalableData.pol_1" origin=easyblend_origin scale=1
[viewport] id=vp_1 x=0 y=0 width=2560 height=1600 projection=proj_easyblend_3
[viewport] id=vp_2 x=2560 y=0 width=2560 height=1600 projection=proj_easyblend_4
[viewport] id=vp_3 x=0 y=0 width=2560 height=1600 projection=proj_easyblend_1
[viewport] id=vp_4 x=2560 y=0 width=2560 height=1600 projection=proj_easyblend_2
[camera] id=camera_static loc="X=0,Y=0,Z=0.0"
[scene_node] id=cave_origin loc="X=0,Y=0,Z=0" rot="P=0,Y=0,R=0"
[scene_node] id=wand loc="X=0,Y=0,Z=1"
[scene_node] id=easyblend_origin loc="X=0,Y=0,Z=0" rot="P=0,Y=0,R=0"
[general] swap_sync_policy=1 ue4_input_sync_policy=1
[network] cln_conn_tries_amount=10 cln_conn_retry_delay=1000 game_start_timeout=30000 barrier_wait_timeout=5000
[custom] SampleArg1=SampleVal1 SampleArg2=SampleVal2
```

采用简单策略以及双显示器设置的完整配置文件示例

```
[info] version="23"
[cluster_node] id="node_left" addr="127.0.0.1" window="wnd_left" master="true"
[cluster_node] id="node_right" addr="127.0.0.1" window="wnd_right"
[window] id="wnd_left" fullscreen="true" viewports="vp_left" WinX="0" WinY="0" ResX="2560" ResY="1440"
[window] id="wnd_right" fullscreen="true" viewports="vp_right" WinX="2560" WinY="0" ResX="2560" ResY="1440"
[projection] id="proj_left" type="simple" screen="scr_left"
[projection] id="proj_right" type="simple" screen="scr_right"
[screen] id="scr_left" loc="X=1.5,Y=-.8889,Z=0" rot="P=0,Y=0,R=0" size="X=1.7778,Y=1.0"
[screen] id="scr_right" loc="X=1.5,Y=.8889,Z=0" rot="P=0,Y=0,R=0" size="X=1.7778,Y=1.0"
[viewport] id="vp_left" x="0" y="0" width="2560" height="1440" projection="proj_left" camera="camera_left"
[viewport] id="vp_right" x="0" y="0" width="2560" height="1440" projection="proj_right" camera="camera_right"
[camera] id=camera_left loc="X=0,Y=0,Z=0.0"
[camera] id=camera_right loc="X=0,Y=0,Z=0.0"
[scene_node] id=cave_origin loc="X=0,Y=0,Z=0" rot="P=0,Y=0,R=0"
[scene_node] id=wand loc="X=0,Y=0,Z=1"
[scene_node] id=proj_origin loc="X=0,Y=0,Z=0" rot="P=0,Y=0,R=0"
[general] swap_sync_policy=1 ue4_input_sync_policy=1
[network] cln_conn_tries_amount=30000 cln_conn_retry_delay=500 game_start_timeout=30000000 barrier_wait_timeout=50000000
[custom] SampleArg1=SampleVal1 SampleArg2=SampleVal2
```

nDisplay 启动器和 nDisplay 监听器

这些应用程序会与 nDisplay 插件一起提供，以便在计算机阵列上部署项目。

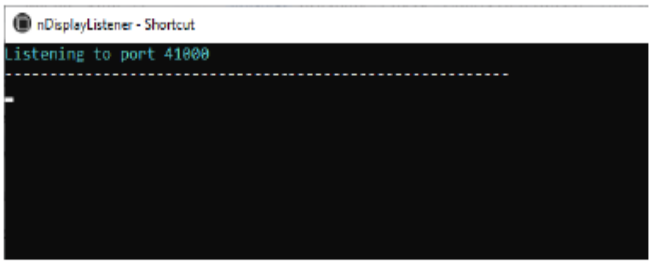


图 18：nDisplay 监听器的软件界面截图

监听器

nDisplay 监听器是一种安装在主机（主计算机和所有群集节点计算机）上的极简应用程序，能够接收各种远程命令，例如使用路径和参数列表启动现有项目，或终止当前项目。

启动器

启动器可以同时为当前可用的一组计算机启动多台投影仪，这些计算机需要同时在后台上运行监听器。启动器可以在本地网络中的任意一台计算机或笔记本计算机上运行。

运行启动器时，请指定以下内容：

- 应用程序路径
- 用于描述群集网络、显示设备拓扑结构和其他所需设置的配置文件
- 其他一些可选设置，例如立体设置选项、项目变量或命令行参数

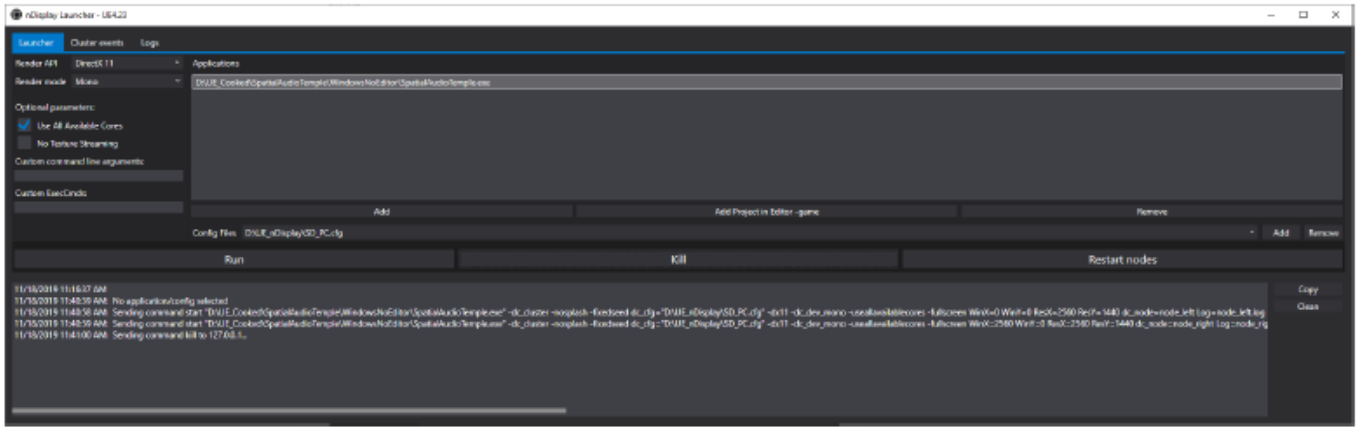


图 19：nDisplay 启动器的软件界面截图

虚幻引擎项目相关的注意事项

假设显示群集已经正确设置，则在项目中启用 nDisplay 会相当简单。多数情况下，无需更改 UE4 场景中的任何内容即可直接使用 nDisplay。但在少数情况下，你需要对设置进行小幅度修改。

nDisplay 主要可用以下两类使用情形：

- **静态系统**——*静态*一词在这里指 nDisplay 的摄像机系统不会对外界影响做出反应，例如用户与显示设备的交互行为。（注意，在这种情况下，UE4 摄像机可能是静态的，也可能是动态的。）在这种常见的使用情况中，由于不需要特定的追踪，你通常可以使用特定的配置文件来运行项目，而无需任何修改。nDisplay 会在 UE4 场景中自动生成一个 DisplayClusterRootActor，它会跟随当前活跃的 UE4 摄像机，无需任何偏移或修正。
- **头部追踪系统**——如果系统中涉及头部跟踪系统（例如 ART、Vicon），请使用配置文件中指定的 VRPN 服务器跟踪系统。多数 CAVE 系统都是以这种方法实现的。在这类使用情况中，位置和方向会自动进行实时调整，以便显示画面在逻辑上与用户的视点保持一致。

无论何种情况，都应该避免为了让 UE4 摄像机符合用户相对于显示画面的实际视点而对摄像机进行手动调整。通常，项目中会有许多摄像机（例如用于游戏玩法、动画效果和过场动画），所以很难手动将所有摄像机调整到正确效果。

如需调整 UE4 的摄像机视图，建议你首先将 DisplayClusterRootActor 添加到项目中的某个位置，并且让 DisplayClusterRootComponent 应用本地偏移量和旋转度。此处所做的调整只会应用到 nDisplay 的摄像机系统上，而不会对 UE4 项目中的主摄像机逻辑产生实质性影响。

另一种极为简便且不会产生实质性影响的做法是，直接将偏移和旋转度变化应用到 nDisplay 配置文件中的“[camera]”一行。

下一步/未来展望

Epic Games 致力于不断改善 nDisplay 技术，我们对其展示出的无限可能性感到激动不已。我们从业界和内部团队得到的大量反馈极大地拓宽了我们的视野，并让我们更加深刻地认识到了这项技术的未来前景。

随着 nDisplay 及其生态系统的不断发展，一些崭新的行业机遇正在不断涌现，例如光雕投影、以及能以任意尺寸、形状和分辨率显示的画幕。在即将发布的最新 nDisplay 版本中，我们将更加注重用户体验，并解决剩余的确定性问题。我们的长期目标是提供一个成熟的框架，让所有的虚幻引擎项目都能投射渲染内容，而且可以使用任何游戏功能，包括物理模拟和游戏逻辑等功能。换句话说，我们的目标是提供一个能够兼容所有虚幻引擎特性的确定性系统。

在当前和即将发布的版本中，我们的计划是让所有的虚幻引擎项目都能在尽可能不修改或不中断的情况下运行并显示内容。虽然这会涉及一些复杂的底层操作，但我们为最终用户提出的理念不会改变：

“启用 nDisplay，定义屏幕拓扑结构，投射实时渲染内容。”

虽然我们的计划可能出现一些意外情况或临时调整，但总的来说，我们希望用户可以专注应用程序的创意开发和逻辑开发，让 nDisplay 负责处理渲染画面的分配和显示工作。

关于本文

作者

Sevan Dalkian

贡献人员

Sébastien Miglio

Sébastien Lozé

Simon Tourangeau

Vitalii Boiko

Andrey Yamashev

编辑

Michele Bousquet

排版

Jung Kwak